4.10.1

* 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)
* [User Manual Home](http://docs.google.com/userguide/userguide.html)
* [DSL Reference Home](http://docs.google.com/dsl/)
* [Release Notes](http://docs.google.com/release-notes.html)
  + [Properties](http://docs.google.com/org.gradle.api.Project.html#N14D65)
    - [announce plugin](http://docs.google.com/org.gradle.api.Project.html#N14EE4)
    - [application plugin](http://docs.google.com/org.gradle.api.Project.html#N14F08)
    - [checkstyle plugin](http://docs.google.com/org.gradle.api.Project.html#N14F42)
    - [codenarc plugin](http://docs.google.com/org.gradle.api.Project.html#N14F66)
    - [distribution plugin](http://docs.google.com/org.gradle.api.Project.html#N14F8A)
    - [ear plugin](http://docs.google.com/org.gradle.api.Project.html#N14FAE)
    - [eclipse plugin](http://docs.google.com/org.gradle.api.Project.html#N14FDF)
    - [findbugs plugin](http://docs.google.com/org.gradle.api.Project.html#N15003)
    - [idea plugin](http://docs.google.com/org.gradle.api.Project.html#N15027)
    - [jacoco plugin](http://docs.google.com/org.gradle.api.Project.html#N1504B)
    - [java plugin](http://docs.google.com/org.gradle.api.Project.html#N1506F)
    - [jdepend plugin](http://docs.google.com/org.gradle.api.Project.html#N15116)
    - [maven plugin](http://docs.google.com/org.gradle.api.Project.html#N1513A)
    - [pmd plugin](http://docs.google.com/org.gradle.api.Project.html#N15162)
    - [project-report plugin](http://docs.google.com/org.gradle.api.Project.html#N15186)
    - [publishing plugin](http://docs.google.com/org.gradle.api.Project.html#N151AE)
    - [signing plugin](http://docs.google.com/org.gradle.api.Project.html#N151D2)
    - [visual-studio plugin](http://docs.google.com/org.gradle.api.Project.html#N151F6)
    - [war plugin](http://docs.google.com/org.gradle.api.Project.html#N1521A)
  + [Methods](http://docs.google.com/org.gradle.api.Project.html#N15242)
    - [ear plugin](http://docs.google.com/org.gradle.api.Project.html#N1558B)
    - [java plugin](http://docs.google.com/org.gradle.api.Project.html#N155BF)
    - [maven plugin](http://docs.google.com/org.gradle.api.Project.html#N15602)
    - [osgi plugin](http://docs.google.com/org.gradle.api.Project.html#N15645)
  + [Script blocks](http://docs.google.com/org.gradle.api.Project.html#N15688)
    - [announce plugin](http://docs.google.com/org.gradle.api.Project.html#N156E6)
    - [checkstyle plugin](http://docs.google.com/org.gradle.api.Project.html#N1570A)
    - [codenarc plugin](http://docs.google.com/org.gradle.api.Project.html#N1572E)
    - [distribution plugin](http://docs.google.com/org.gradle.api.Project.html#N15752)
    - [ear plugin](http://docs.google.com/org.gradle.api.Project.html#N15776)
    - [eclipse plugin](http://docs.google.com/org.gradle.api.Project.html#N15795)
    - [findbugs plugin](http://docs.google.com/org.gradle.api.Project.html#N157B9)
    - [idea plugin](http://docs.google.com/org.gradle.api.Project.html#N157DD)
    - [jacoco plugin](http://docs.google.com/org.gradle.api.Project.html#N15801)
    - [java plugin](http://docs.google.com/org.gradle.api.Project.html#N15825)
    - [jdepend plugin](http://docs.google.com/org.gradle.api.Project.html#N15852)
    - [pmd plugin](http://docs.google.com/org.gradle.api.Project.html#N15876)
    - [publishing plugin](http://docs.google.com/org.gradle.api.Project.html#N1589A)
    - [signing plugin](http://docs.google.com/org.gradle.api.Project.html#N158BE)
    - [visual-studio plugin](http://docs.google.com/org.gradle.api.Project.html#N158E2)

### Build script blocks

* [allprojects { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:allprojects(groovy.lang.Closure))
* [artifacts { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:artifacts(groovy.lang.Closure))
* [buildscript { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildscript(groovy.lang.Closure))
* [configurations { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configurations(groovy.lang.Closure))
* [dependencies { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:dependencies(groovy.lang.Closure))
* [repositories { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:repositories(groovy.lang.Closure))
* [sourceSets { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sourceSets(groovy.lang.Closure))
* [subprojects { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:subprojects(groovy.lang.Closure))
* [publishing { }](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:publishing(groovy.lang.Closure))

### Core types

* [Project](http://docs.google.com/org.gradle.api.Project.html)
* [Task](http://docs.google.com/org.gradle.api.Task.html)
* [Gradle](http://docs.google.com/org.gradle.api.invocation.Gradle.html)
* [Settings](http://docs.google.com/org.gradle.api.initialization.Settings.html)
* [IncludedBuild](http://docs.google.com/org.gradle.api.initialization.IncludedBuild.html)
* [Script](http://docs.google.com/org.gradle.api.Script.html)
* [JavaToolChain](http://docs.google.com/org.gradle.jvm.toolchain.JavaToolChain.html)
* [SourceSet](http://docs.google.com/org.gradle.api.tasks.SourceSet.html)
* [SourceSetOutput](http://docs.google.com/org.gradle.api.tasks.SourceSetOutput.html)
* [SourceDirectorySet](http://docs.google.com/org.gradle.api.file.SourceDirectorySet.html)
* [IncrementalTaskInputs](http://docs.google.com/org.gradle.api.tasks.incremental.IncrementalTaskInputs.html)
* [Configuration](http://docs.google.com/org.gradle.api.artifacts.Configuration.html)
* [ResolutionStrategy](http://docs.google.com/org.gradle.api.artifacts.ResolutionStrategy.html)
* [ArtifactResolutionQuery](http://docs.google.com/org.gradle.api.artifacts.query.ArtifactResolutionQuery.html)
* [ComponentSelection](http://docs.google.com/org.gradle.api.artifacts.ComponentSelection.html)
* [ComponentSelectionRules](http://docs.google.com/org.gradle.api.artifacts.ComponentSelectionRules.html)
* [ExtensionAware](http://docs.google.com/org.gradle.api.plugins.ExtensionAware.html)
* [ExtraPropertiesExtension](http://docs.google.com/org.gradle.api.plugins.ExtraPropertiesExtension.html)
* [PluginDependenciesSpec](http://docs.google.com/org.gradle.plugin.use.PluginDependenciesSpec.html)
* [PluginDependencySpec](http://docs.google.com/org.gradle.plugin.use.PluginDependencySpec.html)
* [PluginManagementSpec](http://docs.google.com/org.gradle.plugin.management.PluginManagementSpec.html)
* [ResourceHandler](http://docs.google.com/org.gradle.api.resources.ResourceHandler.html)
* [TextResourceFactory](http://docs.google.com/org.gradle.api.resources.TextResourceFactory.html)

### Publishing types

* [PublishingExtension](http://docs.google.com/org.gradle.api.publish.PublishingExtension.html)
* [IvyPublication](http://docs.google.com/org.gradle.api.publish.ivy.IvyPublication.html)
* [IvyArtifact](http://docs.google.com/org.gradle.api.publish.ivy.IvyArtifact.html)
* [IvyArtifactSet](http://docs.google.com/org.gradle.api.publish.ivy.IvyArtifactSet.html)
* [IvyModuleDescriptorSpec](http://docs.google.com/org.gradle.api.publish.ivy.IvyModuleDescriptorSpec.html)
* [IvyModuleDescriptorAuthor](http://docs.google.com/org.gradle.api.publish.ivy.IvyModuleDescriptorAuthor.html)
* [IvyModuleDescriptorLicense](http://docs.google.com/org.gradle.api.publish.ivy.IvyModuleDescriptorLicense.html)
* [IvyModuleDescriptorDescription](http://docs.google.com/org.gradle.api.publish.ivy.IvyModuleDescriptorDescription.html)
* [MavenPublication](http://docs.google.com/org.gradle.api.publish.maven.MavenPublication.html)
* [MavenArtifact](http://docs.google.com/org.gradle.api.publish.maven.MavenArtifact.html)
* [MavenArtifactSet](http://docs.google.com/org.gradle.api.publish.maven.MavenArtifactSet.html)
* [MavenPom](http://docs.google.com/org.gradle.api.publish.maven.MavenPom.html)
* [MavenPomCiManagement](http://docs.google.com/org.gradle.api.publish.maven.MavenPomCiManagement.html)
* [MavenPomContributor](http://docs.google.com/org.gradle.api.publish.maven.MavenPomContributor.html)
* [MavenPomContributorSpec](http://docs.google.com/org.gradle.api.publish.maven.MavenPomContributorSpec.html)
* [MavenPomDeveloper](http://docs.google.com/org.gradle.api.publish.maven.MavenPomDeveloper.html)
* [MavenPomDeveloperSpec](http://docs.google.com/org.gradle.api.publish.maven.MavenPomDeveloperSpec.html)
* [MavenPomDistributionManagement](http://docs.google.com/org.gradle.api.publish.maven.MavenPomDistributionManagement.html)
* [MavenPomIssueManagement](http://docs.google.com/org.gradle.api.publish.maven.MavenPomIssueManagement.html)
* [MavenPomLicense](http://docs.google.com/org.gradle.api.publish.maven.MavenPomLicense.html)
* [MavenPomLicenseSpec](http://docs.google.com/org.gradle.api.publish.maven.MavenPomLicenseSpec.html)
* [MavenPomMailingList](http://docs.google.com/org.gradle.api.publish.maven.MavenPomMailingList.html)
* [MavenPomMailingListSpec](http://docs.google.com/org.gradle.api.publish.maven.MavenPomMailingListSpec.html)
* [MavenPomOrganization](http://docs.google.com/org.gradle.api.publish.maven.MavenPomOrganization.html)
* [MavenPomRelocation](http://docs.google.com/org.gradle.api.publish.maven.MavenPomRelocation.html)
* [MavenPomScm](http://docs.google.com/org.gradle.api.publish.maven.MavenPomScm.html)

### Container types

* [TaskContainer](http://docs.google.com/org.gradle.api.tasks.TaskContainer.html)
* [ConfigurationContainer](http://docs.google.com/org.gradle.api.artifacts.ConfigurationContainer.html)
* [RepositoryHandler](http://docs.google.com/org.gradle.api.artifacts.dsl.RepositoryHandler.html)
* [DependencyHandler](http://docs.google.com/org.gradle.api.artifacts.dsl.DependencyHandler.html)
* [ArtifactHandler](http://docs.google.com/org.gradle.api.artifacts.dsl.ArtifactHandler.html)

### Build Cache types

* [BuildCacheConfiguration](http://docs.google.com/org.gradle.caching.configuration.BuildCacheConfiguration.html)
* [DirectoryBuildCache](http://docs.google.com/org.gradle.caching.local.DirectoryBuildCache.html)
* [HttpBuildCache](http://docs.google.com/org.gradle.caching.http.HttpBuildCache.html)

### Input Normalization types

* [InputNormalizationHandler](http://docs.google.com/org.gradle.normalization.InputNormalizationHandler.html)
* [InputNormalization](http://docs.google.com/org.gradle.normalization.InputNormalization.html)
* [RuntimeClasspathNormalization](http://docs.google.com/org.gradle.normalization.RuntimeClasspathNormalization.html)

### Help Task types

* [TaskReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.TaskReportTask.html)
* [ProjectReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.ProjectReportTask.html)
* [DependencyReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.DependencyReportTask.html)
* [DependencyInsightReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.DependencyInsightReportTask.html)
* [PropertyReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.PropertyReportTask.html)
* [ComponentReport](http://docs.google.com/org.gradle.api.reporting.components.ComponentReport.html)
* [DependentComponentsReport](http://docs.google.com/org.gradle.api.reporting.dependents.DependentComponentsReport.html)
* [ModelReport](http://docs.google.com/org.gradle.api.reporting.model.ModelReport.html)

### Task types

* [AntlrTask](http://docs.google.com/org.gradle.api.plugins.antlr.AntlrTask.html)
* [BuildEnvironmentReportTask](http://docs.google.com/org.gradle.api.tasks.diagnostics.BuildEnvironmentReportTask.html)
* [Checkstyle](http://docs.google.com/org.gradle.api.plugins.quality.Checkstyle.html)
* [CodeNarc](http://docs.google.com/org.gradle.api.plugins.quality.CodeNarc.html)
* [CompareGradleBuilds](http://docs.google.com/org.gradle.api.plugins.buildcomparison.gradle.CompareGradleBuilds.html)
* [Copy](http://docs.google.com/org.gradle.api.tasks.Copy.html)
* [CreateStartScripts](http://docs.google.com/org.gradle.jvm.application.tasks.CreateStartScripts.html)
* [Delete](http://docs.google.com/org.gradle.api.tasks.Delete.html)
* [Ear](http://docs.google.com/org.gradle.plugins.ear.Ear.html)
* [Exec](http://docs.google.com/org.gradle.api.tasks.Exec.html)
* [FindBugs](http://docs.google.com/org.gradle.api.plugins.quality.FindBugs.html)
* [GenerateIvyDescriptor](http://docs.google.com/org.gradle.api.publish.ivy.tasks.GenerateIvyDescriptor.html)
* [GenerateMavenPom](http://docs.google.com/org.gradle.api.publish.maven.tasks.GenerateMavenPom.html)
* [GenerateBuildDashboard](http://docs.google.com/org.gradle.api.reporting.GenerateBuildDashboard.html)
* [GradleBuild](http://docs.google.com/org.gradle.api.tasks.GradleBuild.html)
* [GroovyCompile](http://docs.google.com/org.gradle.api.tasks.compile.GroovyCompile.html)
* [Groovydoc](http://docs.google.com/org.gradle.api.tasks.javadoc.Groovydoc.html)
* [HtmlDependencyReportTask](http://docs.google.com/org.gradle.api.reporting.dependencies.HtmlDependencyReportTask.html)
* [JacocoReport](http://docs.google.com/org.gradle.testing.jacoco.tasks.JacocoReport.html)
* [JacocoMerge](http://docs.google.com/org.gradle.testing.jacoco.tasks.JacocoMerge.html)
* [JacocoCoverageVerification](http://docs.google.com/org.gradle.testing.jacoco.tasks.JacocoCoverageVerification.html)
* [Jar](http://docs.google.com/org.gradle.api.tasks.bundling.Jar.html)
* [JavaCompile](http://docs.google.com/org.gradle.api.tasks.compile.JavaCompile.html)
* [Javadoc](http://docs.google.com/org.gradle.api.tasks.javadoc.Javadoc.html)
* [JavaExec](http://docs.google.com/org.gradle.api.tasks.JavaExec.html)
* [JDepend](http://docs.google.com/org.gradle.api.plugins.quality.JDepend.html)
* [Pmd](http://docs.google.com/org.gradle.api.plugins.quality.Pmd.html)
* [PublishToIvyRepository](http://docs.google.com/org.gradle.api.publish.ivy.tasks.PublishToIvyRepository.html)
* [PublishToMavenRepository](http://docs.google.com/org.gradle.api.publish.maven.tasks.PublishToMavenRepository.html)
* [ScalaCompile](http://docs.google.com/org.gradle.api.tasks.scala.ScalaCompile.html)
* [ScalaDoc](http://docs.google.com/org.gradle.api.tasks.scala.ScalaDoc.html)
* [InitBuild](http://docs.google.com/org.gradle.buildinit.tasks.InitBuild.html)
* [Sign](http://docs.google.com/org.gradle.plugins.signing.Sign.html)
* [Sync](http://docs.google.com/org.gradle.api.tasks.Sync.html)
* [Tar](http://docs.google.com/org.gradle.api.tasks.bundling.Tar.html)
* [AbstractTestTask](http://docs.google.com/org.gradle.api.tasks.testing.AbstractTestTask.html)
* [Test](http://docs.google.com/org.gradle.api.tasks.testing.Test.html)
* [TestReport](http://docs.google.com/org.gradle.api.tasks.testing.TestReport.html)
* [Upload](http://docs.google.com/org.gradle.api.tasks.Upload.html)
* [War](http://docs.google.com/org.gradle.api.tasks.bundling.War.html)
* [Wrapper](http://docs.google.com/org.gradle.api.tasks.wrapper.Wrapper.html)
* [WriteProperties](http://docs.google.com/org.gradle.api.tasks.WriteProperties.html)
* [Zip](http://docs.google.com/org.gradle.api.tasks.bundling.Zip.html)

### Reporting types

* [CustomizableHtmlReport](http://docs.google.com/org.gradle.api.reporting.CustomizableHtmlReport.html)
* [SingleFileReport](http://docs.google.com/org.gradle.api.reporting.SingleFileReport.html)
* [DirectoryReport](http://docs.google.com/org.gradle.api.reporting.DirectoryReport.html)
* [FindBugsXmlReport](http://docs.google.com/org.gradle.api.plugins.quality.FindBugsXmlReport.html)
* [Report](http://docs.google.com/org.gradle.api.reporting.Report.html)
* [Reporting](http://docs.google.com/org.gradle.api.reporting.Reporting.html)
* [ReportContainer](http://docs.google.com/org.gradle.api.reporting.ReportContainer.html)
* [ReportingExtension](http://docs.google.com/org.gradle.api.reporting.ReportingExtension.html)

### Eclipse/IDEA model types

* [EclipseModel](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseModel.html)
* [EclipseProject](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseProject.html)
* [EclipseClasspath](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseClasspath.html)
* [EclipseJdt](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseJdt.html)
* [EclipseWtp](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseWtp.html)
* [EclipseWtpComponent](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseWtpComponent.html)
* [EclipseWtpFacet](http://docs.google.com/org.gradle.plugins.ide.eclipse.model.EclipseWtpFacet.html)
* [IdeaModel](http://docs.google.com/org.gradle.plugins.ide.idea.model.IdeaModel.html)
* [IdeaProject](http://docs.google.com/org.gradle.plugins.ide.idea.model.IdeaProject.html)
* [IdeaModule](http://docs.google.com/org.gradle.plugins.ide.idea.model.IdeaModule.html)
* [IdeaWorkspace](http://docs.google.com/org.gradle.plugins.ide.idea.model.IdeaWorkspace.html)
* [XmlFileContentMerger](http://docs.google.com/org.gradle.plugins.ide.api.XmlFileContentMerger.html)
* [FileContentMerger](http://docs.google.com/org.gradle.plugins.ide.api.FileContentMerger.html)

### Eclipse/IDEA task types

* [GenerateEclipseProject](http://docs.google.com/org.gradle.plugins.ide.eclipse.GenerateEclipseProject.html)
* [GenerateEclipseClasspath](http://docs.google.com/org.gradle.plugins.ide.eclipse.GenerateEclipseClasspath.html)
* [GenerateEclipseJdt](http://docs.google.com/org.gradle.plugins.ide.eclipse.GenerateEclipseJdt.html)
* [GenerateEclipseWtpComponent](http://docs.google.com/org.gradle.plugins.ide.eclipse.GenerateEclipseWtpComponent.html)
* [GenerateEclipseWtpFacet](http://docs.google.com/org.gradle.plugins.ide.eclipse.GenerateEclipseWtpFacet.html)
* [GenerateIdeaModule](http://docs.google.com/org.gradle.plugins.ide.idea.GenerateIdeaModule.html)
* [GenerateIdeaProject](http://docs.google.com/org.gradle.plugins.ide.idea.GenerateIdeaProject.html)
* [GenerateIdeaWorkspace](http://docs.google.com/org.gradle.plugins.ide.idea.GenerateIdeaWorkspace.html)

### Native software types

* [PrebuiltLibrary](http://docs.google.com/org.gradle.nativeplatform.PrebuiltLibrary.html)
* [PrebuiltSharedLibraryBinary](http://docs.google.com/org.gradle.nativeplatform.PrebuiltSharedLibraryBinary.html)
* [PrebuiltStaticLibraryBinary](http://docs.google.com/org.gradle.nativeplatform.PrebuiltStaticLibraryBinary.html)
* [NativeComponentSpec](http://docs.google.com/org.gradle.nativeplatform.NativeComponentSpec.html)
* [NativeExecutableSpec](http://docs.google.com/org.gradle.nativeplatform.NativeExecutableSpec.html)
* [NativeLibrarySpec](http://docs.google.com/org.gradle.nativeplatform.NativeLibrarySpec.html)
* [NativeTestSuiteSpec](http://docs.google.com/org.gradle.nativeplatform.test.NativeTestSuiteSpec.html)
* [CUnitTestSuiteSpec](http://docs.google.com/org.gradle.nativeplatform.test.cunit.CUnitTestSuiteSpec.html)
* [GoogleTestTestSuiteSpec](http://docs.google.com/org.gradle.nativeplatform.test.googletest.GoogleTestTestSuiteSpec.html)
* [NativeBinarySpec](http://docs.google.com/org.gradle.nativeplatform.NativeBinarySpec.html)
* [NativeExecutableBinarySpec](http://docs.google.com/org.gradle.nativeplatform.NativeExecutableBinarySpec.html)
* [NativeLibraryBinarySpec](http://docs.google.com/org.gradle.nativeplatform.NativeLibraryBinarySpec.html)
* [SharedLibraryBinarySpec](http://docs.google.com/org.gradle.nativeplatform.SharedLibraryBinarySpec.html)
* [StaticLibraryBinarySpec](http://docs.google.com/org.gradle.nativeplatform.StaticLibraryBinarySpec.html)
* [NativeTestSuiteBinarySpec](http://docs.google.com/org.gradle.nativeplatform.test.NativeTestSuiteBinarySpec.html)
* [CUnitTestSuiteBinarySpec](http://docs.google.com/org.gradle.nativeplatform.test.cunit.CUnitTestSuiteBinarySpec.html)
* [GoogleTestTestSuiteBinarySpec](http://docs.google.com/org.gradle.nativeplatform.test.googletest.GoogleTestTestSuiteBinarySpec.html)
* [NativePlatform](http://docs.google.com/org.gradle.nativeplatform.platform.NativePlatform.html)
* [BuildType](http://docs.google.com/org.gradle.nativeplatform.BuildType.html)
* [Flavor](http://docs.google.com/org.gradle.nativeplatform.Flavor.html)
* [Gcc](http://docs.google.com/org.gradle.nativeplatform.toolchain.Gcc.html)
* [Clang](http://docs.google.com/org.gradle.nativeplatform.toolchain.Clang.html)
* [VisualCpp](http://docs.google.com/org.gradle.nativeplatform.toolchain.VisualCpp.html)
* [AssemblerSourceSet](http://docs.google.com/org.gradle.language.assembler.AssemblerSourceSet.html)
* [CSourceSet](http://docs.google.com/org.gradle.language.c.CSourceSet.html)
* [CppSourceSet](http://docs.google.com/org.gradle.language.cpp.CppSourceSet.html)
* [ObjectiveCSourceSet](http://docs.google.com/org.gradle.language.objectivec.ObjectiveCSourceSet.html)
* [ObjectiveCppSourceSet](http://docs.google.com/org.gradle.language.objectivecpp.ObjectiveCppSourceSet.html)
* [WindowsResourceSet](http://docs.google.com/org.gradle.language.rc.WindowsResourceSet.html)
* [VisualStudioProject](http://docs.google.com/org.gradle.ide.visualstudio.VisualStudioProject.html)
* [VisualStudioSolution](http://docs.google.com/org.gradle.ide.visualstudio.VisualStudioSolution.html)
* [NativeExecutable](http://docs.google.com/org.gradle.nativeplatform.NativeExecutable.html)
* [NativeLibrary](http://docs.google.com/org.gradle.nativeplatform.NativeLibrary.html)
* [NativeBinary](http://docs.google.com/org.gradle.nativeplatform.NativeBinary.html)
* [NativeExecutableBinary](http://docs.google.com/org.gradle.nativeplatform.NativeExecutableBinary.html)
* [SharedLibraryBinary](http://docs.google.com/org.gradle.nativeplatform.SharedLibraryBinary.html)
* [StaticLibraryBinary](http://docs.google.com/org.gradle.nativeplatform.StaticLibraryBinary.html)

### Native component task types

* [CppCompile](http://docs.google.com/org.gradle.language.cpp.tasks.CppCompile.html)
* [CCompile](http://docs.google.com/org.gradle.language.c.tasks.CCompile.html)
* [Assemble](http://docs.google.com/org.gradle.language.assembler.tasks.Assemble.html)
* [ObjectiveCCompile](http://docs.google.com/org.gradle.language.objectivec.tasks.ObjectiveCCompile.html)
* [ObjectiveCppCompile](http://docs.google.com/org.gradle.language.objectivecpp.tasks.ObjectiveCppCompile.html)
* [WindowsResourceCompile](http://docs.google.com/org.gradle.language.rc.tasks.WindowsResourceCompile.html)
* [LinkExecutable](http://docs.google.com/org.gradle.nativeplatform.tasks.LinkExecutable.html)
* [LinkSharedLibrary](http://docs.google.com/org.gradle.nativeplatform.tasks.LinkSharedLibrary.html)
* [CreateStaticLibrary](http://docs.google.com/org.gradle.nativeplatform.tasks.CreateStaticLibrary.html)
* [InstallExecutable](http://docs.google.com/org.gradle.nativeplatform.tasks.InstallExecutable.html)
* [RunTestExecutable](http://docs.google.com/org.gradle.nativeplatform.test.tasks.RunTestExecutable.html)

Project

**Table of Contents**

[Lifecycle](http://docs.google.com/org.gradle.api.Project.html#N14C11)[Tasks](http://docs.google.com/org.gradle.api.Project.html#N14C52)[Dependencies](http://docs.google.com/org.gradle.api.Project.html#N14C73)[Multi-project Builds](http://docs.google.com/org.gradle.api.Project.html#N14CA0)[Plugins](http://docs.google.com/org.gradle.api.Project.html#N14CA5)[Properties](http://docs.google.com/org.gradle.api.Project.html#N14CB6)[Properties](http://docs.google.com/org.gradle.api.Project.html#N14D65)[Methods](http://docs.google.com/org.gradle.api.Project.html#N15242)[Script blocks](http://docs.google.com/org.gradle.api.Project.html#N15688)[Property details](http://docs.google.com/org.gradle.api.Project.html#N15906)[Method details](http://docs.google.com/org.gradle.api.Project.html#N15E33)[Script block details](http://docs.google.com/org.gradle.api.Project.html#N16899)

| API Documentation: | [Project](http://docs.google.com/javadoc/org/gradle/api/Project.html) |
| --- | --- |

This interface is the main API you use to interact with Gradle from your build file. From a Project, you have programmatic access to all of Gradle's features.

Lifecycle

There is a one-to-one relationship between a Project and a build.gradle file. During build initialisation, Gradle assembles a Project object for each project which is to participate in the build, as follows:

* Create a [Settings](http://docs.google.com/dsl/org.gradle.api.initialization.Settings.html) instance for the build.
* Evaluate the settings.gradle script, if present, against the [Settings](http://docs.google.com/dsl/org.gradle.api.initialization.Settings.html) object to configure it.
* Use the configured [Settings](http://docs.google.com/dsl/org.gradle.api.initialization.Settings.html) object to create the hierarchy of Project instances.
* Finally, evaluate each Project by executing its build.gradle file, if present, against the project. The projects are evaluated in breadth-wise order, such that a project is evaluated before its child projects. This order can be overridden by calling [Project.evaluationDependsOnChildren()](http://docs.google.com/javadoc/org/gradle/api/Project.html#evaluationDependsOnChildren--) or by adding an explicit evaluation dependency using [Project.evaluationDependsOn(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:evaluationDependsOn(java.lang.String)).

Tasks

A project is essentially a collection of [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) objects. Each task performs some basic piece of work, such as compiling classes, or running unit tests, or zipping up a WAR file. You add tasks to a project using one of the create() methods on [TaskContainer](http://docs.google.com/dsl/org.gradle.api.tasks.TaskContainer.html), such as [TaskContainer.create(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.tasks.TaskContainer.html#org.gradle.api.tasks.TaskContainer:create(java.lang.String)). You can locate existing tasks using one of the lookup methods on [TaskContainer](http://docs.google.com/dsl/org.gradle.api.tasks.TaskContainer.html), such as [TaskCollection.getByName(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.tasks.TaskCollection.html#org.gradle.api.tasks.TaskCollection:getByName(java.lang.String)).

Dependencies

A project generally has a number of dependencies it needs in order to do its work. Also, a project generally produces a number of artifacts, which other projects can use. Those dependencies are grouped in configurations, and can be retrieved and uploaded from repositories. You use the [ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html) returned by [Project.getConfigurations()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:configurations) method to manage the configurations. The [DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html) returned by [Project.getDependencies()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:dependencies) method to manage the dependencies. The [ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) returned by [Project.getArtifacts()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:artifacts) method to manage the artifacts. The [RepositoryHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.RepositoryHandler.html) returned by [Project.getRepositories()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:repositories) method to manage the repositories.

Multi-project Builds

Projects are arranged into a hierarchy of projects. A project has a name, and a fully qualified path which uniquely identifies it in the hierarchy.

Plugins

Plugins can be used to modularise and reuse project configuration. Plugins can be applied using the [PluginAware.apply(java.util.Map)](http://docs.google.com/dsl/org.gradle.api.plugins.PluginAware.html#org.gradle.api.plugins.PluginAware:apply(java.util.Map)) method, or by using the [PluginDependenciesSpec](http://docs.google.com/dsl/org.gradle.plugin.use.PluginDependenciesSpec.html) plugins script block.

Properties

Gradle executes the project's build file against the Project instance to configure the project. Any property or method which your script uses is delegated through to the associated Project object. This means, that you can use any of the methods and properties on the Project interface directly in your script.

For example:

defaultTasks('some-task') // Delegates to Project.defaultTasks()  
reportsDir = file('reports') // Delegates to Project.file() and the Java Plugin

You can also access the Project instance using the project property. This can make the script clearer in some cases. For example, you could use project.name rather than name to access the project's name.

A project has 5 property 'scopes', which it searches for properties. You can access these properties by name in your build file, or by calling the project's [Project.property(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:property(java.lang.String)) method. The scopes are:

* The Project object itself. This scope includes any property getters and setters declared by the Project implementation class. For example, [Project.getRootProject()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:rootProject) is accessible as the rootProject property. The properties of this scope are readable or writable depending on the presence of the corresponding getter or setter method.
* The *extra* properties of the project. Each project maintains a map of extra properties, which can contain any arbitrary name -> value pair. Once defined, the properties of this scope are readable and writable. See [extra properties](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.extraproperties) for more details.
* The *extensions* added to the project by the plugins. Each extension is available as a read-only property with the same name as the extension.
* The *convention* properties added to the project by the plugins. A plugin can add properties and methods to a project through the project's [Convention](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) object. The properties of this scope may be readable or writable, depending on the convention objects.
* The tasks of the project. A task is accessible by using its name as a property name. The properties of this scope are read-only. For example, a task called compile is accessible as the compile property.
* The extra properties and convention properties are inherited from the project's parent, recursively up to the root project. The properties of this scope are read-only.

When reading a property, the project searches the above scopes in order, and returns the value from the first scope it finds the property in. If not found, an exception is thrown. See [Project.property(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:property(java.lang.String)) for more details.

When writing a property, the project searches the above scopes in order, and sets the property in the first scope it finds the property in. If not found, an exception is thrown. See [Project.setProperty(java.lang.String, java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:setProperty(java.lang.String,%20java.lang.Object)) for more details.

Extra Properties

All extra properties must be defined through the "ext" namespace. Once an extra property has been defined, it is available directly on the owning object (in the below case the Project, Task, and sub-projects respectively) and can be read and updated. Only the initial declaration that needs to be done via the namespace.

project.ext.prop1 = "foo"  
task doStuff {  
 ext.prop2 = "bar"  
}  
subprojects { ext.${prop3} = false }

Reading extra properties is done through the "ext" or through the owning object.

ext.isSnapshot = version.endsWith("-SNAPSHOT")  
if (isSnapshot) {  
 // do snapshot stuff  
}

Dynamic Methods

A project has 5 method 'scopes', which it searches for methods:

* The Project object itself.
* The build file. The project searches for a matching method declared in the build file.
* The *extensions* added to the project by the plugins. Each extension is available as a method which takes a closure or [Action](http://docs.google.com/javadoc/org/gradle/api/Action.html) as a parameter.
* The *convention* methods added to the project by the plugins. A plugin can add properties and method to a project through the project's [Convention](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) object.
* The tasks of the project. A method is added for each task, using the name of the task as the method name and taking a single closure or [Action](http://docs.google.com/javadoc/org/gradle/api/Action.html) parameter. The method calls the [Task.configure(groovy.lang.Closure)](http://docs.google.com/javadoc/org/gradle/api/Task.html#configure-groovy.lang.Closure-) method for the associated task with the provided closure. For example, if the project has a task called compile, then a method is added with the following signature: void compile(Closure configureClosure).
* The methods of the parent project, recursively up to the root project.
* A property of the project whose value is a closure. The closure is treated as a method and called with the provided parameters. The property is located as described above.

Properties

| Property | Description |
| --- | --- |
| [allprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:allprojects) | The set containing this project and its subprojects. |
| [ant](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:ant) | The AntBuilder for this project. You can use this in your build file to execute ant tasks. See example below. |
| [artifacts](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:artifacts) | Returns a handler for assigning artifacts produced by the project to configurations. |
| [buildDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildDir) | The build directory of this project. The build directory is the directory which all artifacts are generated into. The default value for the build directory is *projectDir*/build |
| [buildFile](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildFile) | The build script for this project. |
| [buildscript](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildscript) | The build script handler for this project. You can use this handler to query details about the build script for this project, and manage the classpath used to compile and execute the project's build script. |
| [childProjects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:childProjects) | The direct children of this project. |
| [configurations](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configurations) | The configurations of this project. |
| [convention](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:convention) | The [Convention](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) for this project. |
| [defaultTasks](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:defaultTasks) | The names of the default tasks of this project. These are used when no tasks names are provided when starting the build. |
| [dependencies](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:dependencies) | The dependency handler of this project. The returned dependency handler instance can be used for adding new dependencies. For accessing already declared dependencies, the configurations can be used. |
| [description](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:description) | The description of this project, if any. |
| [extensions](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:extensions) | Allows adding DSL extensions to the project. Useful for plugin authors. |
| [gradle](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:gradle) | The [Gradle](http://docs.google.com/dsl/org.gradle.api.invocation.Gradle.html) invocation which this project belongs to. |
| [group](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:group) | The group of this project. Gradle always uses the toString() value of the group. The group defaults to the path with dots as separators. |
| [logger](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:logger) | The logger for this project. You can use this in your build file to write log messages. |
| [logging](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:logging) | The [LoggingManager](http://docs.google.com/javadoc/org/gradle/api/logging/LoggingManager.html) which can be used to receive logging and to control the standard output/error capture for this project's build script. By default, System.out is redirected to the Gradle logging system at the QUIET log level, and System.err is redirected at the ERROR log level. |
| [name](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:name) | The name of this project. The project's name is not necessarily unique within a project hierarchy. You should use the [Project.getPath()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:path) method for a unique identifier for the project. |
| [normalization](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:normalization) | Incubating  Provides access to configuring input normalization. |
| [parent](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:parent) | The parent project of this project, if any. |
| [path](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:path) | The path of this project. The path is the fully qualified name of the project. |
| [pluginManager](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pluginManager) | Incubating  The plugin manager for this plugin aware object. |
| [plugins](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:plugins) | The container of plugins that have been applied to this object. |
| [project](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:project) | Returns this project. This method is useful in build files to explicitly access project properties and methods. For example, using project.name can express your intent better than using name. This method also allows you to access project properties from a scope where the property may be hidden, such as, for example, from a method or closure. |
| [projectDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:projectDir) | The directory containing the project build file. |
| [properties](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:properties) | The properties of this project. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for details of the properties which are available for a project. |
| [repositories](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:repositories) | Returns a handler to create repositories which are used for retrieving dependencies and uploading artifacts produced by the project. |
| [resources](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:resources) | Provides access to resource-specific utility methods, for example factory methods that create various resources. |
| [rootDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:rootDir) | The root directory of this project. The root directory is the project directory of the root project. |
| [rootProject](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:rootProject) | The root project for the hierarchy that this project belongs to. In the case of a single-project build, this method returns this project. |
| [state](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:state) | The evaluation state of this project. You can use this to access information about the evaluation of this project, such as whether it has failed. |
| [status](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:status) | The status of this project. Gradle always uses the toString() value of the status. The status defaults to release. |
| [subprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:subprojects) | The set containing the subprojects of this project. |
| [tasks](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:tasks) | The tasks of this project. |
| [version](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:version) | The version of this project. Gradle always uses the toString() value of the version. The version defaults to unspecified. |

Properties added by the announce plugin

| Property | Description |
| --- | --- |
| [announce](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:announce) | The [AnnouncePluginExtension](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) added by the announce plugin. |

Properties added by the application plugin

| Property | Description |
| --- | --- |
| [applicationDefaultJvmArgs](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:applicationDefaultJvmArgs) | Array of string arguments to pass to the JVM when running the application |
| [applicationDistribution](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:applicationDistribution) | The specification of the contents of the distribution. |
| [applicationName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:applicationName) | The name of the application. |
| [mainClassName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:mainClassName) | The fully qualified name of the application's main class. |

Properties added by the checkstyle plugin

| Property | Description |
| --- | --- |
| [checkstyle](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:checkstyle) | The [CheckstyleExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) added by the checkstyle plugin. |

Properties added by the codenarc plugin

| Property | Description |
| --- | --- |
| [codenarc](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:codenarc) | The [CodeNarcExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) added by the codenarc plugin. |

Properties added by the distribution plugin

| Property | Description |
| --- | --- |
| [distributions](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:distributions) | The [DistributionContainer](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) added by the distribution plugin. |

Properties added by the ear plugin

| Property | Description |
| --- | --- |
| [appDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:appDirName) | The name of the application directory, relative to the project directory. Default is "src/main/application". |
| [deploymentDescriptor](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:deploymentDescriptor) | A custom deployment descriptor configuration. Default is an "application.xml" with sensible defaults. |
| [libDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:libDirName) | The name of the library directory in the EAR file. Default is "lib". |

Properties added by the eclipse plugin

| Property | Description |
| --- | --- |
| [eclipse](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:eclipse) | The [EclipseModel](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) added by the eclipse plugin. |

Properties added by the findbugs plugin

| Property | Description |
| --- | --- |
| [findbugs](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:findbugs) | The [FindBugsExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) added by the findbugs plugin. |

Properties added by the idea plugin

| Property | Description |
| --- | --- |
| [idea](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:idea) | The [IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html) added by the idea plugin. |

Properties added by the jacoco plugin

| Property | Description |
| --- | --- |
| [jacoco](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jacoco) | The [JacocoPluginExtension](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) added by the jacoco plugin. |

Properties added by the java plugin

| Property | Description |
| --- | --- |
| [archivesBaseName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:archivesBaseName) | The base name to use for archive files. |
| [distsDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:distsDir) | The directory to generate TAR and ZIP archives into. |
| [distsDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:distsDirName) | The name for the distributions directory. This in interpreted relative to the project' build directory. |
| [docsDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:docsDir) | Returns a file pointing to the root directory supposed to be used for all docs. |
| [docsDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:docsDirName) | The name of the docs directory. Can be a name or a path relative to the build dir. |
| [libsDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:libsDir) | The directory to generate JAR and WAR archives into. |
| [libsDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:libsDirName) | The name for the libs directory. This in interpreted relative to the project' build directory. |
| [reporting](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:reporting) | The [ReportingExtension](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) added by the java plugin. |
| [sourceCompatibility](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sourceCompatibility) | The source compatibility used for compiling Java sources. |
| [sourceSets](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sourceSets) | The source sets container. |
| [targetCompatibility](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:targetCompatibility) | The target compatibility used for compiling Java sources. |
| [testReportDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:testReportDir) | Returns a file pointing to the root directory to be used for reports. |
| [testReportDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:testReportDirName) | The name of the test reports directory. Can be a name or a path relative to [ReportingExtension.getBaseDir()](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html#org.gradle.api.reporting.ReportingExtension:baseDir). |
| [testResultsDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:testResultsDir) | Returns a file pointing to the root directory of the test results. |
| [testResultsDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:testResultsDirName) | The name of the test results directory. Can be a name or a path relative to the build dir. |

Properties added by the jdepend plugin

| Property | Description |
| --- | --- |
| [jdepend](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jdepend) | The [JDependExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) added by the jdepend plugin. |

Properties added by the maven plugin

| Property | Description |
| --- | --- |
| [conf2ScopeMappings](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:conf2ScopeMappings) | The set of rules for how to map Gradle dependencies to Maven scopes. |
| [mavenPomDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:mavenPomDir) | The directory to generate Maven POMs into. |

Properties added by the pmd plugin

| Property | Description |
| --- | --- |
| [pmd](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pmd) | The [PmdExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) added by the pmd plugin. |

Properties added by the project-report plugin

| Property | Description |
| --- | --- |
| [projectReportDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:projectReportDir) | The directory to generate the project reports into. |
| [projectReportDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:projectReportDirName) | The name of the directory to generate the project reports into, relative to the project's reports dir. |

Properties added by the publishing plugin

| Property | Description |
| --- | --- |
| [publishing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:publishing) | The [PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) added by the publishing plugin. |

Properties added by the signing plugin

| Property | Description |
| --- | --- |
| [signing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:signing) | The [SigningExtension](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) added by the signing plugin. |

Properties added by the visual-studio plugin

| Property | Description |
| --- | --- |
| [visualStudio](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:visualStudio) | The [VisualStudioRootExtension](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) added by the visual-studio plugin. |

Properties added by the war plugin

| Property | Description |
| --- | --- |
| [webAppDir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:webAppDir) | The web application directory. |
| [webAppDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:webAppDirName) | The name of the web application directory, relative to the project directory. |

Methods

| Method | Description |
| --- | --- |
| [absoluteProjectPath](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:absoluteProjectPath(java.lang.String))(path) | Converts a name to an absolute project path, resolving names relative to this project. |
| [afterEvaluate](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:afterEvaluate(groovy.lang.Closure))(closure) | Adds a closure to be called immediately after this project has been evaluated. The project is passed to the closure as a parameter. Such a listener gets notified when the build file belonging to this project has been executed. A parent project may for example add such a listener to its child project. Such a listener can further configure those child projects based on the state of the child projects after their build files have been run. |
| [afterEvaluate](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:afterEvaluate(org.gradle.api.Action))(action) | Adds an action to execute immediately after this project is evaluated. |
| [allprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:allprojects(org.gradle.api.Action))(action) | Configures this project and each of its sub-projects. |
| [ant](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:ant(org.gradle.api.Action))(configureAction) | Executes the given action against the AntBuilder for this project. You can use this in your build file to execute ant tasks. See example in javadoc for [Project.getAnt()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:ant) |
| [apply](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:apply(groovy.lang.Closure))(closure) | Applies zero or more plugins or scripts. |
| [apply](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:apply(java.util.Map))(options) | Applies a plugin or script, using the given options provided as a map. Does nothing if the plugin has already been applied. |
| [apply](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:apply(org.gradle.api.Action))(action) | Applies zero or more plugins or scripts. |
| [artifacts](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:artifacts(org.gradle.api.Action))(configureAction) | Configures the published artifacts for this project. |
| [beforeEvaluate](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:beforeEvaluate(groovy.lang.Closure))(closure) | Adds a closure to be called immediately before this project is evaluated. The project is passed to the closure as a parameter. |
| [beforeEvaluate](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:beforeEvaluate(org.gradle.api.Action))(action) | Adds an action to execute immediately before this project is evaluated. |
| [configure](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configure(java.lang.Iterable,%20groovy.lang.Closure))(objects, configureClosure) | Configures a collection of objects via a closure. This is equivalent to calling [Project.configure(java.lang.Object, groovy.lang.Closure)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:configure(java.lang.Object,%20groovy.lang.Closure)) for each of the given objects. |
| [configure](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configure(java.lang.Iterable,%20org.gradle.api.Action))(objects, configureAction) | Configures a collection of objects via an action. |
| [configure](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configure(java.lang.Object,%20groovy.lang.Closure))(object, configureClosure) | Configures an object via a closure, with the closure's delegate set to the supplied object. This way you don't have to specify the context of a configuration statement multiple times. |
| [container](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:container(java.lang.Class))(type) | Creates a container for managing named objects of the specified type. The specified type must have a public constructor which takes the name as a String parameter. |
| [container](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:container(java.lang.Class,%20groovy.lang.Closure))(type, factoryClosure) | Creates a container for managing named objects of the specified type. The given closure is used to create object instances. The name of the instance to be created is passed as a parameter to the closure. |
| [container](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:container(java.lang.Class,%20org.gradle.api.NamedDomainObjectFactory))(type, factory) | Creates a container for managing named objects of the specified type. The given factory is used to create object instances. |
| [copy](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:copy(groovy.lang.Closure))(closure) | Copies the specified files. The given closure is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to copy the files. Example: |
| [copy](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:copy(org.gradle.api.Action))(action) | Copies the specified files. The given action is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to copy the files. |
| [copySpec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:copySpec())() | Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive. |
| [copySpec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:copySpec(groovy.lang.Closure))(closure) | Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive. The given closure is used to configure the [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) before it is returned by this method. |
| [copySpec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:copySpec(org.gradle.api.Action))(action) | Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive. The given action is used to configure the [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) before it is returned by this method. |
| [delete](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:delete(java.lang.Object%5B%5D))(paths) | Deletes files and directories. |
| [delete](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:delete(org.gradle.api.Action))(action) | Deletes the specified files. The given action is used to configure a [DeleteSpec](http://docs.google.com/javadoc/org/gradle/api/file/DeleteSpec.html), which is then used to delete the files. |
| [evaluationDependsOn](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:evaluationDependsOn(java.lang.String))(path) | Declares that this project has an evaluation dependency on the project with the given path. |
| [exec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:exec(groovy.lang.Closure))(closure) | Executes an external command. The closure configures a [ExecSpec](http://docs.google.com/javadoc/org/gradle/process/ExecSpec.html). |
| [exec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:exec(org.gradle.api.Action))(action) | Executes an external command. |
| [file](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object))(path) | Resolves a file path relative to the project directory of this project. This method converts the supplied path based on its type: |
| [file](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object,%20org.gradle.api.PathValidation))(path, validation) | Resolves a file path relative to the project directory of this project and validates it using the given scheme. See [PathValidation](http://docs.google.com/javadoc/org/gradle/api/PathValidation.html) for the list of possible validations. |
| [fileTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:fileTree(java.lang.Object))(baseDir) | Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). |
| [fileTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:fileTree(java.lang.Object,%20groovy.lang.Closure))(baseDir, configureClosure) | Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). The closure will be used to configure the new file tree. The file tree is passed to the closure as its delegate. Example: |
| [fileTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:fileTree(java.lang.Object,%20org.gradle.api.Action))(baseDir, configureAction) | Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). The action will be used to configure the new file tree. Example: |
| [fileTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:fileTree(java.util.Map))(args) | Creates a new ConfigurableFileTree using the provided map of arguments. The map will be applied as properties on the new file tree. Example: |
| [files](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object,%20groovy.lang.Closure))(paths, configureClosure) | Creates a new ConfigurableFileCollection using the given paths. The paths are evaluated as per [Project.files(java.lang.Object[])](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object%5B%5D)). The file collection is configured using the given closure. The file collection is passed to the closure as its delegate. Example: |
| [files](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object,%20org.gradle.api.Action))(paths, configureAction) | Creates a new ConfigurableFileCollection using the given paths. The paths are evaluated as per [Project.files(java.lang.Object[])](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object%5B%5D)). The file collection is configured using the given action. Example: |
| [files](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object%5B%5D))(paths) | Returns a [ConfigurableFileCollection](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html) containing the given files. You can pass any of the following types to this method: |
| [findProject](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:findProject(java.lang.String))(path) | Locates a project by path. If the path is relative, it is interpreted relative to this project. |
| [findProperty](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:findProperty(java.lang.String))(propertyName) | Returns the value of the given property or null if not found. This method locates a property as follows: |
| [getAllTasks](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:getAllTasks(boolean))(recursive) | Returns a map of the tasks contained in this project, and optionally its subprojects. |
| [getTasksByName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:getTasksByName(java.lang.String,%20boolean))(name, recursive) | Returns the set of tasks with the given name contained in this project, and optionally its subprojects. *NOTE:* This is an expensive operation since it requires all projects to be configured. |
| [hasProperty](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:hasProperty(java.lang.String))(propertyName) | Determines if this project has the given property. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for details of the properties which are available for a project. |
| [javaexec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:javaexec(groovy.lang.Closure))(closure) | Executes a Java main class. The closure configures a [JavaExecSpec](http://docs.google.com/javadoc/org/gradle/process/JavaExecSpec.html). |
| [javaexec](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:javaexec(org.gradle.api.Action))(action) | Executes an external Java process. |
| [mkdir](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:mkdir(java.lang.Object))(path) | Creates a directory and returns a file pointing to it. |
| [normalization](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:normalization(org.gradle.api.Action))(configuration) | Incubating  Configures input normalization. |
| [project](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:project(java.lang.String))(path) | Locates a project by path. If the path is relative, it is interpreted relative to this project. |
| [project](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:project(java.lang.String,%20groovy.lang.Closure))(path, configureClosure) | Locates a project by path and configures it using the given closure. If the path is relative, it is interpreted relative to this project. The target project is passed to the closure as the closure's delegate. |
| [project](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:project(java.lang.String,%20org.gradle.api.Action))(path, configureAction) | Locates a project by path and configures it using the given action. If the path is relative, it is interpreted relative to this project. |
| [property](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:property(java.lang.Class))(clazz) | Deprecated  Incubating  Creates a PropertyState implementation based on the provided class. |
| [property](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:property(java.lang.String))(propertyName) | Returns the value of the given property. This method locates a property as follows: |
| [relativePath](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:relativePath(java.lang.Object))(path) | Returns the relative path from the project directory to the given path. The given path object is (logically) resolved as described for [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)), from which a relative path is calculated. |
| [relativeProjectPath](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:relativeProjectPath(java.lang.String))(path) | Converts a name to a project path relative to this project. |
| [setProperty](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:setProperty(java.lang.String,%20java.lang.Object))(name, value) | Sets a property of this project. This method searches for a property with the given name in the following locations, and sets the property on the first location where it finds the property. |
| [subprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:subprojects(org.gradle.api.Action))(action) | Configures the sub-projects of this project |
| [sync](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sync(org.gradle.api.Action))(action) | Synchronizes the contents of a destination directory with some source directories and files. The given action is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to synchronize the files. |
| [tarTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:tarTree(java.lang.Object))(tarPath) | Creates a new FileTree which contains the contents of the given TAR file. The given tarPath path can be: |
| [task](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:task(java.lang.String))(name) | Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Calling this method is equivalent to calling [Project.task(java.util.Map, java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String)) with an empty options map. |
| [task](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:task(java.lang.String,%20groovy.lang.Closure))(name, configureClosure) | Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given closure is executed to configure the task. |
| [task](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:task(java.lang.String,%20org.gradle.api.Action))(name, configureAction) | Incubating  Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given action is executed to configure the task. |
| [task](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String))(args, name) | Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. A map of creation options can be passed to this method to control how the task is created. The following options are available: |
| [task](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String,%20groovy.lang.Closure))(args, name, configureClosure) | Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given closure is executed to configure the task. A map of creation options can be passed to this method to control how the task is created. See [Project.task(java.util.Map, java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String)) for the available options. |
| [uri](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:uri(java.lang.Object))(path) | Resolves a file path to a URI, relative to the project directory of this project. Evaluates the provided path object as described for [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)), with the exception that any URI scheme is supported, not just 'file:' URIs. |
| [zipTree](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:zipTree(java.lang.Object))(zipPath) | Creates a new FileTree which contains the contents of the given ZIP file. The given zipPath path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). You can combine this method with the [Project.copy(groovy.lang.Closure)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:copy(groovy.lang.Closure)) method to unzip a ZIP file. |

Methods added by the ear plugin

| Method | Description |
| --- | --- |
| [appDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:appDirName(java.lang.String))(appDirName) | Allows changing the application directory. Default is "src/main/application". |
| [deploymentDescriptor](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:deploymentDescriptor(org.gradle.api.Action))(configureAction) | Configures the deployment descriptor for this EAR archive. |
| [libDirName](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:libDirName(java.lang.String))(libDirName) | Allows changing the library directory in the EAR file. Default is "lib". |

Methods added by the java plugin

| Method | Description |
| --- | --- |
| [manifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:manifest())() | Creates a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html). |
| [manifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:manifest(groovy.lang.Closure))(closure) | Creates and configures a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html). The given closure configures the new manifest instance before it is returned. |
| [manifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:manifest(org.gradle.api.Action))(action) | Creates and configures a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html). |

Methods added by the maven plugin

| Method | Description |
| --- | --- |
| [pom](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pom())() | Creates a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html). |
| [pom](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pom(groovy.lang.Closure))(configureClosure) | Creates and configures a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html). The given closure is executed to configure the new POM instance. |
| [pom](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pom(org.gradle.api.Action))(configureAction) | Creates and configures a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html). The given action is executed to configure the new POM instance. |

Methods added by the osgi plugin

| Method | Description |
| --- | --- |
| [osgiManifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:osgiManifest())() | Creates a new instance of [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html). The returned object is preconfigured with: |
| [osgiManifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:osgiManifest(groovy.lang.Closure))(closure) | Creates and configures a new instance of an [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html) . The closure configures the new manifest instance before it is returned. |
| [osgiManifest](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:osgiManifest(org.gradle.api.Action))(action) | Creates and configures a new instance of an [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html). The action configures the new manifest instance before it is returned. |

Script blocks

| Block | Description |
| --- | --- |
| [allprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:allprojects(groovy.lang.Closure)) | Configures this project and each of its sub-projects. |
| [ant](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:ant(groovy.lang.Closure)) | Executes the given closure against the AntBuilder for this project. You can use this in your build file to execute ant tasks. The AntBuild is passed to the closure as the closure's delegate. See example in javadoc for [Project.getAnt()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:ant) |
| [artifacts](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:artifacts(groovy.lang.Closure)) | Configures the published artifacts for this project. |
| [buildscript](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildscript(groovy.lang.Closure)) | Configures the build script classpath for this project. |
| [configurations](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configurations(groovy.lang.Closure)) | Configures the dependency configurations for this project. |
| [dependencies](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:dependencies(groovy.lang.Closure)) | Configures the dependencies for this project. |
| [repositories](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:repositories(groovy.lang.Closure)) | Configures the repositories for this project. |
| [subprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:subprojects(groovy.lang.Closure)) | Configures the sub-projects of this project. |

Script blocks added by the announce plugin

| Block | Description |
| --- | --- |
| [announce](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:announce(groovy.lang.Closure)) | Configures the [AnnouncePluginExtension](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) added by the announce plugin. |

Script blocks added by the checkstyle plugin

| Block | Description |
| --- | --- |
| [checkstyle](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:checkstyle(groovy.lang.Closure)) | Configures the [CheckstyleExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) added by the checkstyle plugin. |

Script blocks added by the codenarc plugin

| Block | Description |
| --- | --- |
| [codenarc](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:codenarc(groovy.lang.Closure)) | Configures the [CodeNarcExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) added by the codenarc plugin. |

Script blocks added by the distribution plugin

| Block | Description |
| --- | --- |
| [distributions](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:distributions(groovy.lang.Closure)) | Configures the [DistributionContainer](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) added by the distribution plugin. |

Script blocks added by the ear plugin

| Block | Description |
| --- | --- |
| [deploymentDescriptor](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:deploymentDescriptor(groovy.lang.Closure)) | Configures the deployment descriptor for this EAR archive. |

Script blocks added by the eclipse plugin

| Block | Description |
| --- | --- |
| [eclipse](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:eclipse(groovy.lang.Closure)) | Configures the [EclipseModel](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) added by the eclipse plugin. |

Script blocks added by the findbugs plugin

| Block | Description |
| --- | --- |
| [findbugs](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:findbugs(groovy.lang.Closure)) | Configures the [FindBugsExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) added by the findbugs plugin. |

Script blocks added by the idea plugin

| Block | Description |
| --- | --- |
| [idea](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:idea(groovy.lang.Closure)) | Configures the [IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html) added by the idea plugin. |

Script blocks added by the jacoco plugin

| Block | Description |
| --- | --- |
| [jacoco](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jacoco(groovy.lang.Closure)) | Configures the [JacocoPluginExtension](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) added by the jacoco plugin. |

Script blocks added by the java plugin

| Block | Description |
| --- | --- |
| [reporting](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:reporting(groovy.lang.Closure)) | Configures the [ReportingExtension](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) added by the java plugin. |
| [sourceSets](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sourceSets(groovy.lang.Closure)) | Configures the source sets of this project. |

Script blocks added by the jdepend plugin

| Block | Description |
| --- | --- |
| [jdepend](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jdepend(groovy.lang.Closure)) | Configures the [JDependExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) added by the jdepend plugin. |

Script blocks added by the pmd plugin

| Block | Description |
| --- | --- |
| [pmd](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pmd(groovy.lang.Closure)) | Configures the [PmdExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) added by the pmd plugin. |

Script blocks added by the publishing plugin

| Block | Description |
| --- | --- |
| [publishing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:publishing(groovy.lang.Closure)) | Configures the [PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) added by the publishing plugin. |

Script blocks added by the signing plugin

| Block | Description |
| --- | --- |
| [signing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:signing(groovy.lang.Closure)) | Configures the [SigningExtension](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) added by the signing plugin. |

Script blocks added by the visual-studio plugin

| Block | Description |
| --- | --- |
| [visualStudio](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:visualStudio(groovy.lang.Closure)) | Configures the [VisualStudioRootExtension](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) added by the visual-studio plugin. |

Property details

[**Set**](http://download.oracle.com/javase/8/docs/api/java/util/Set.html)<[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> allprojects (read-only)

The set containing this project and its subprojects.

[**AntBuilder**](http://docs.google.com/javadoc/org/gradle/api/AntBuilder.html) ant (read-only)

The AntBuilder for this project. You can use this in your build file to execute ant tasks. See example below.

task printChecksum {  
 doLast {  
 ant {  
 //using ant checksum task to store the file checksum in the checksumOut ant property  
 checksum(property: 'checksumOut', file: 'someFile.txt')  
  
 //we can refer to the ant property created by checksum task:  
 println "The checksum is: " + checksumOut  
 }  
  
 //we can refer to the ant property later as well:  
 println "I just love to print checksums: " + ant.checksumOut  
 }  
}

Consider following example of ant target:

<target name='printChecksum'>  
 <checksum property='checksumOut'>  
 <fileset dir='.'>  
 <include name='agile.txt'/>  
 </fileset>  
 </checksum>  
 <echo>The checksum is: ${checksumOut}</echo>  
</target>

Here's how it would look like in gradle. Observe how the ant XML is represented in groovy by the ant builder

task printChecksum {  
 doLast {  
 ant {  
 checksum(property: 'checksumOut') {  
 fileset(dir: '.') {  
 include name: 'agile1.txt'  
 }  
 }  
 }  
 logger.lifecycle("The checksum is $ant.checksumOut")  
 }  
}

[**ArtifactHandler**](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) artifacts (read-only)

Returns a handler for assigning artifacts produced by the project to configurations.

Examples:

See docs for [ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html)

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) buildDir

The build directory of this project. The build directory is the directory which all artifacts are generated into. The default value for the build directory is *projectDir*/build

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) buildFile (read-only)

The build script for this project.

If the file exists, it will be evaluated against this project when this project is configured.

[**ScriptHandler**](http://docs.google.com/javadoc/org/gradle/api/initialization/dsl/ScriptHandler.html) buildscript (read-only)

The build script handler for this project. You can use this handler to query details about the build script for this project, and manage the classpath used to compile and execute the project's build script.

[**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> childProjects (read-only)

The direct children of this project.

[**ConfigurationContainer**](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html) configurations (read-only)

The configurations of this project.

Examples:

See docs for [ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html)

[**Convention**](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) convention (read-only)

The [Convention](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) for this project.

You can access this property in your build file using convention. You can also access the properties and methods of the convention object as if they were properties and methods of this project. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

[**List**](http://download.oracle.com/javase/8/docs/api/java/util/List.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html)> defaultTasks

The names of the default tasks of this project. These are used when no tasks names are provided when starting the build.

[**DependencyHandler**](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html) dependencies (read-only)

The dependency handler of this project. The returned dependency handler instance can be used for adding new dependencies. For accessing already declared dependencies, the configurations can be used.

Examples:

See docs for [DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html)

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) description

The description of this project, if any.

[**ExtensionContainer**](http://docs.google.com/javadoc/org/gradle/api/plugins/ExtensionContainer.html) extensions (read-only)

Allows adding DSL extensions to the project. Useful for plugin authors.

[**Gradle**](http://docs.google.com/dsl/org.gradle.api.invocation.Gradle.html) gradle (read-only)

The [Gradle](http://docs.google.com/dsl/org.gradle.api.invocation.Gradle.html) invocation which this project belongs to.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) group

The group of this project. Gradle always uses the toString() value of the group. The group defaults to the path with dots as separators.

[**Logger**](http://docs.google.com/javadoc/org/gradle/api/logging/Logger.html) logger (read-only)

The logger for this project. You can use this in your build file to write log messages.

[**LoggingManager**](http://docs.google.com/javadoc/org/gradle/api/logging/LoggingManager.html) logging (read-only)

The [LoggingManager](http://docs.google.com/javadoc/org/gradle/api/logging/LoggingManager.html) which can be used to receive logging and to control the standard output/error capture for this project's build script. By default, System.out is redirected to the Gradle logging system at the QUIET log level, and System.err is redirected at the ERROR log level.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name (read-only)

The name of this project. The project's name is not necessarily unique within a project hierarchy. You should use the [Project.getPath()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:path) method for a unique identifier for the project.

[**InputNormalizationHandler**](http://docs.google.com/dsl/org.gradle.normalization.InputNormalizationHandler.html) normalization (read-only)

Note: This property is [incubating](http://docs.google.com/userguide/feature_lifecycle.html) and may change in a future version of Gradle.

Provides access to configuring input normalization.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) parent (read-only)

The parent project of this project, if any.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path (read-only)

The path of this project. The path is the fully qualified name of the project.

[**PluginManager**](http://docs.google.com/dsl/org.gradle.api.plugins.PluginManager.html) pluginManager (read-only)

Note: This property is [incubating](http://docs.google.com/userguide/feature_lifecycle.html) and may change in a future version of Gradle.

The plugin manager for this plugin aware object.

[**PluginContainer**](http://docs.google.com/javadoc/org/gradle/api/plugins/PluginContainer.html) plugins (read-only)

The container of plugins that have been applied to this object.

While not deprecated, it is preferred to use the methods of this interface or the [PluginAware.getPluginManager()](http://docs.google.com/dsl/org.gradle.api.plugins.PluginAware.html#org.gradle.api.plugins.PluginAware:pluginManager) than use the plugin container.

Use one of the 'apply' methods on this interface or on the [PluginAware.getPluginManager()](http://docs.google.com/dsl/org.gradle.api.plugins.PluginAware.html#org.gradle.api.plugins.PluginAware:pluginManager) to apply plugins instead of applying via the plugin container.

Use [PluginManager.hasPlugin(java.lang.String)](http://docs.google.com/dsl/org.gradle.api.plugins.PluginManager.html#org.gradle.api.plugins.PluginManager:hasPlugin(java.lang.String)) or similar to query for the application of plugins instead of doing so via the plugin container.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) project (read-only)

Returns this project. This method is useful in build files to explicitly access project properties and methods. For example, using project.name can express your intent better than using name. This method also allows you to access project properties from a scope where the property may be hidden, such as, for example, from a method or closure.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) projectDir (read-only)

The directory containing the project build file.

[**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), ?> properties (read-only)

The properties of this project. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for details of the properties which are available for a project.

[**RepositoryHandler**](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.RepositoryHandler.html) repositories (read-only)

Returns a handler to create repositories which are used for retrieving dependencies and uploading artifacts produced by the project.

[**ResourceHandler**](http://docs.google.com/dsl/org.gradle.api.resources.ResourceHandler.html) resources (read-only)

Provides access to resource-specific utility methods, for example factory methods that create various resources.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) rootDir (read-only)

The root directory of this project. The root directory is the project directory of the root project.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) rootProject (read-only)

The root project for the hierarchy that this project belongs to. In the case of a single-project build, this method returns this project.

[**ProjectState**](http://docs.google.com/javadoc/org/gradle/api/ProjectState.html) state (read-only)

The evaluation state of this project. You can use this to access information about the evaluation of this project, such as whether it has failed.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) status

The status of this project. Gradle always uses the toString() value of the status. The status defaults to release.

The status of the project is only relevant, if you upload libraries together with a module descriptor. The status specified here, will be part of this module descriptor.

[**Set**](http://download.oracle.com/javase/8/docs/api/java/util/Set.html)<[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> subprojects (read-only)

The set containing the subprojects of this project.

[**TaskContainer**](http://docs.google.com/dsl/org.gradle.api.tasks.TaskContainer.html) tasks (read-only)

The tasks of this project.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) version

The version of this project. Gradle always uses the toString() value of the version. The version defaults to unspecified.

[**AnnouncePluginExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) announce (read-only)

The [AnnouncePluginExtension](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) added by the announce plugin.

[**Iterable**](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html)> applicationDefaultJvmArgs

Array of string arguments to pass to the JVM when running the application

[**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) applicationDistribution

The specification of the contents of the distribution.

Use this [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) to include extra files/resource in the application distribution.

apply plugin: 'application'  
  
applicationDistribution.from("some/dir") {  
 include "\*.txt"  
}

Note that the application plugin pre configures this spec to; include the contents of "src/dist", copy the application start scripts into the "bin" directory, and copy the built jar and its dependencies into the "lib" directory.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) applicationName

The name of the application.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) mainClassName

The fully qualified name of the application's main class.

[**CheckstyleExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) checkstyle (read-only)

The [CheckstyleExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) added by the checkstyle plugin.

[**CodeNarcExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) codenarc (read-only)

The [CodeNarcExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) added by the codenarc plugin.

[**DistributionContainer**](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) distributions (read-only)

The [DistributionContainer](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) added by the distribution plugin.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) appDirName

The name of the application directory, relative to the project directory. Default is "src/main/application".

[**DeploymentDescriptor**](http://docs.google.com/javadoc/org/gradle/plugins/ear/descriptor/DeploymentDescriptor.html) deploymentDescriptor

A custom deployment descriptor configuration. Default is an "application.xml" with sensible defaults.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) libDirName

The name of the library directory in the EAR file. Default is "lib".

[**EclipseModel**](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) eclipse (read-only)

The [EclipseModel](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) added by the eclipse plugin.

[**FindBugsExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) findbugs (read-only)

The [FindBugsExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) added by the findbugs plugin.

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

The [IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html) added by the idea plugin.

[**JacocoPluginExtension**](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) jacoco (read-only)

The [JacocoPluginExtension](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) added by the jacoco plugin.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) archivesBaseName

The base name to use for archive files.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) distsDir (read-only)

The directory to generate TAR and ZIP archives into.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) distsDirName

The name for the distributions directory. This in interpreted relative to the project' build directory.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) docsDir (read-only)

Returns a file pointing to the root directory supposed to be used for all docs.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) docsDirName

The name of the docs directory. Can be a name or a path relative to the build dir.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) libsDir (read-only)

The directory to generate JAR and WAR archives into.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) libsDirName

The name for the libs directory. This in interpreted relative to the project' build directory.

[**ReportingExtension**](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) reporting (read-only)

The [ReportingExtension](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) added by the java plugin.

[**JavaVersion**](http://docs.google.com/javadoc/org/gradle/api/JavaVersion.html) sourceCompatibility

The source compatibility used for compiling Java sources.

[**SourceSetContainer**](http://docs.google.com/javadoc/org/gradle/api/tasks/SourceSetContainer.html) sourceSets (read-only)

The source sets container.

[**JavaVersion**](http://docs.google.com/javadoc/org/gradle/api/JavaVersion.html) targetCompatibility

The target compatibility used for compiling Java sources.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) testReportDir (read-only)

Returns a file pointing to the root directory to be used for reports.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) testReportDirName

The name of the test reports directory. Can be a name or a path relative to [ReportingExtension.getBaseDir()](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html#org.gradle.api.reporting.ReportingExtension:baseDir).

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) testResultsDir (read-only)

Returns a file pointing to the root directory of the test results.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) testResultsDirName

The name of the test results directory. Can be a name or a path relative to the build dir.

[**JDependExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) jdepend (read-only)

The [JDependExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) added by the jdepend plugin.

[**Conf2ScopeMappingContainer**](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/Conf2ScopeMappingContainer.html) conf2ScopeMappings

The set of rules for how to map Gradle dependencies to Maven scopes.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) mavenPomDir

The directory to generate Maven POMs into.

[**PmdExtension**](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) pmd (read-only)

The [PmdExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) added by the pmd plugin.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) projectReportDir (read-only)

The directory to generate the project reports into.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) projectReportDirName

The name of the directory to generate the project reports into, relative to the project's reports dir.

[**PublishingExtension**](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) publishing (read-only)

The [PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) added by the publishing plugin.

[**SigningExtension**](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) signing (read-only)

The [SigningExtension](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) added by the signing plugin.

[**VisualStudioRootExtension**](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) visualStudio (read-only)

The [VisualStudioRootExtension](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) added by the visual-studio plugin.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) webAppDir (read-only)

The web application directory.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) webAppDirName

The name of the web application directory, relative to the project directory.

Method details

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) absoluteProjectPath([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path)

Converts a name to an absolute project path, resolving names relative to this project.

void afterEvaluate([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Adds a closure to be called immediately after this project has been evaluated. The project is passed to the closure as a parameter. Such a listener gets notified when the build file belonging to this project has been executed. A parent project may for example add such a listener to its child project. Such a listener can further configure those child projects based on the state of the child projects after their build files have been run.

void afterEvaluate([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> action)

Adds an action to execute immediately after this project is evaluated.

void allprojects([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> action)

Configures this project and each of its sub-projects.

This method executes the given [Action](http://docs.google.com/javadoc/org/gradle/api/Action.html) against this project and each of its sub-projects.

[**AntBuilder**](http://docs.google.com/javadoc/org/gradle/api/AntBuilder.html) ant([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**AntBuilder**](http://docs.google.com/javadoc/org/gradle/api/AntBuilder.html)> configureAction)

Executes the given action against the AntBuilder for this project. You can use this in your build file to execute ant tasks. See example in javadoc for [Project.getAnt()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:ant)

void apply([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Applies zero or more plugins or scripts.

The given closure is used to configure an [ObjectConfigurationAction](http://docs.google.com/javadoc/org/gradle/api/plugins/ObjectConfigurationAction.html), which “builds” the plugin application.

This method differs from [PluginAware.apply(java.util.Map)](http://docs.google.com/dsl/org.gradle.api.plugins.PluginAware.html#org.gradle.api.plugins.PluginAware:apply(java.util.Map)) in that it allows methods of the configuration action to be invoked more than once.

void apply([**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), ?> options)

Applies a plugin or script, using the given options provided as a map. Does nothing if the plugin has already been applied.

The given map is applied as a series of method calls to a newly created [ObjectConfigurationAction](http://docs.google.com/javadoc/org/gradle/api/plugins/ObjectConfigurationAction.html). That is, each key in the map is expected to be the name of a method [ObjectConfigurationAction](http://docs.google.com/javadoc/org/gradle/api/plugins/ObjectConfigurationAction.html) and the value to be compatible arguments to that method.

The following options are available:

* from: A script to apply. Accepts any path supported by [Project.uri(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:uri(java.lang.Object)).
* plugin: The id or implementation class of the plugin to apply.
* to: The target delegate object or objects. The default is this plugin aware object. Use this to configure objects other than this object.

void apply([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**ObjectConfigurationAction**](http://docs.google.com/javadoc/org/gradle/api/plugins/ObjectConfigurationAction.html)> action)

Applies zero or more plugins or scripts.

The given closure is used to configure an [ObjectConfigurationAction](http://docs.google.com/javadoc/org/gradle/api/plugins/ObjectConfigurationAction.html), which “builds” the plugin application.

This method differs from [PluginAware.apply(java.util.Map)](http://docs.google.com/dsl/org.gradle.api.plugins.PluginAware.html#org.gradle.api.plugins.PluginAware:apply(java.util.Map)) in that it allows methods of the configuration action to be invoked more than once.

void artifacts([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**ArtifactHandler**](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html)> configureAction)

Configures the published artifacts for this project.

This method executes the given action against the [ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) for this project.

Example:

configurations {  
 //declaring new configuration that will be used to associate with artifacts  
 schema  
}  
  
task schemaJar(type: Jar) {  
 //some imaginary task that creates a jar artifact with the schema  
}  
  
//associating the task that produces the artifact with the configuration  
artifacts {  
 //configuration name and the task:  
 schema schemaJar  
}

void beforeEvaluate([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Adds a closure to be called immediately before this project is evaluated. The project is passed to the closure as a parameter.

void beforeEvaluate([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> action)

Adds an action to execute immediately before this project is evaluated.

[**Iterable**](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html)<?> configure([**Iterable**](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html)<?> objects, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Configures a collection of objects via a closure. This is equivalent to calling [Project.configure(java.lang.Object, groovy.lang.Closure)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:configure(java.lang.Object,%20groovy.lang.Closure)) for each of the given objects.

[**Iterable**](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html)<T> configure([**Iterable**](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html)<T> objects, [**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super T> configureAction)

Configures a collection of objects via an action.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) configure([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) object, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Configures an object via a closure, with the closure's delegate set to the supplied object. This way you don't have to specify the context of a configuration statement multiple times.

Instead of:

MyType myType = new MyType()  
myType.doThis()  
myType.doThat()

you can do:

MyType myType = configure(new MyType()) {  
 doThis()  
 doThat()  
}

The object being configured is also passed to the closure as a parameter, so you can access it explicitly if required:

configure(someObj) { obj -> obj.doThis() }

[**NamedDomainObjectContainer**](http://docs.google.com/dsl/org.gradle.api.NamedDomainObjectContainer.html)<T> container([**Class**](http://download.oracle.com/javase/8/docs/api/java/lang/Class.html)<T> type)

Creates a container for managing named objects of the specified type. The specified type must have a public constructor which takes the name as a String parameter.

All objects *MUST* expose their name as a bean property named "name". The name must be constant for the life of the object.

[**NamedDomainObjectContainer**](http://docs.google.com/dsl/org.gradle.api.NamedDomainObjectContainer.html)<T> container([**Class**](http://download.oracle.com/javase/8/docs/api/java/lang/Class.html)<T> type, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) factoryClosure)

Creates a container for managing named objects of the specified type. The given closure is used to create object instances. The name of the instance to be created is passed as a parameter to the closure.

All objects *MUST* expose their name as a bean property named "name". The name must be constant for the life of the object.

[**NamedDomainObjectContainer**](http://docs.google.com/dsl/org.gradle.api.NamedDomainObjectContainer.html)<T> container([**Class**](http://download.oracle.com/javase/8/docs/api/java/lang/Class.html)<T> type, [**NamedDomainObjectFactory**](http://docs.google.com/javadoc/org/gradle/api/NamedDomainObjectFactory.html)<T> factory)

Creates a container for managing named objects of the specified type. The given factory is used to create object instances.

All objects *MUST* expose their name as a bean property named "name". The name must be constant for the life of the object.

[**WorkResult**](http://docs.google.com/javadoc/org/gradle/api/tasks/WorkResult.html) copy([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Copies the specified files. The given closure is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to copy the files. Example:

copy {  
 from configurations.runtime  
 into 'build/deploy/lib'  
}

Note that CopySpecs can be nested:

copy {  
 into 'build/webroot'  
 exclude '\*\*/.svn/\*\*'  
 from('src/main/webapp') {  
 include '\*\*/\*.jsp'  
 filter(ReplaceTokens, tokens:[copyright:'2009', version:'2.3.1'])  
 }  
 from('src/main/js') {  
 include '\*\*/\*.js'  
 }  
}

[**WorkResult**](http://docs.google.com/javadoc/org/gradle/api/tasks/WorkResult.html) copy([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html)> action)

Copies the specified files. The given action is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to copy the files.

[**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) copySpec()

Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive.

[**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) copySpec([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive. The given closure is used to configure the [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) before it is returned by this method.

def baseSpec = copySpec {  
 from "source"  
 include "\*\*/\*.java"  
}  
  
task copy(type: Copy) {  
 into "target"  
 with baseSpec  
}

[**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) copySpec([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html)> action)

Creates a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) which can later be used to copy files or create an archive. The given action is used to configure the [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html) before it is returned by this method.

boolean delete([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html)... paths)

Deletes files and directories.

This will not follow symlinks. If you need to follow symlinks too use [Project.delete(org.gradle.api.Action)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:delete(org.gradle.api.Action)).

[**WorkResult**](http://docs.google.com/javadoc/org/gradle/api/tasks/WorkResult.html) delete([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**DeleteSpec**](http://docs.google.com/javadoc/org/gradle/api/file/DeleteSpec.html)> action)

Deletes the specified files. The given action is used to configure a [DeleteSpec](http://docs.google.com/javadoc/org/gradle/api/file/DeleteSpec.html), which is then used to delete the files.

Example:

project.delete {  
 delete 'somefile'  
 followSymlinks = true  
}

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) evaluationDependsOn([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path)

Declares that this project has an evaluation dependency on the project with the given path.

[**ExecResult**](http://docs.google.com/javadoc/org/gradle/process/ExecResult.html) exec([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Executes an external command. The closure configures a [ExecSpec](http://docs.google.com/javadoc/org/gradle/process/ExecSpec.html).

[**ExecResult**](http://docs.google.com/javadoc/org/gradle/process/ExecResult.html) exec([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**ExecSpec**](http://docs.google.com/javadoc/org/gradle/process/ExecSpec.html)> action)

Executes an external command.

The given action configures a [ExecSpec](http://docs.google.com/javadoc/org/gradle/process/ExecSpec.html), which is used to launch the process. This method blocks until the process terminates, with its result being returned.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) file([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) path)

Resolves a file path relative to the project directory of this project. This method converts the supplied path based on its type:

* A [CharSequence](http://download.oracle.com/javase/8/docs/api/java/lang/CharSequence.html), including [String](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) or [GString](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/GString.html). Interpreted relative to the project directory. A string that starts with file: is treated as a file URL.
* A [File](http://download.oracle.com/javase/8/docs/api/java/io/File.html). If the file is an absolute file, it is returned as is. Otherwise, the file's path is interpreted relative to the project directory.
* A [Path](http://download.oracle.com/javase/8/docs/api/java/nio/file/Path.html). The path must be associated with the default provider and is treated the same way as an instance of File.
* A [URI](http://download.oracle.com/javase/8/docs/api/java/net/URI.html) or [URL](http://download.oracle.com/javase/8/docs/api/java/net/URL.html). The URL's path is interpreted as the file path. Only file: URLs are supported.
* A [Directory](http://docs.google.com/javadoc/org/gradle/api/file/Directory.html) or [RegularFile](http://docs.google.com/javadoc/org/gradle/api/file/RegularFile.html).
* A [Provider](http://docs.google.com/javadoc/org/gradle/api/provider/Provider.html) of any supported type. The provider's value is resolved recursively.
* A [Closure](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) that returns any supported type. The closure's return value is resolved recursively.
* A [Callable](http://download.oracle.com/javase/8/docs/api/java/util/concurrent/Callable.html) that returns any supported type. The callable's return value is resolved recursively.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) file([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) path, [**PathValidation**](http://docs.google.com/javadoc/org/gradle/api/PathValidation.html) validation)

Resolves a file path relative to the project directory of this project and validates it using the given scheme. See [PathValidation](http://docs.google.com/javadoc/org/gradle/api/PathValidation.html) for the list of possible validations.

[**ConfigurableFileTree**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileTree.html) fileTree([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) baseDir)

Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)).

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

def myTree = fileTree("src")  
myTree.include "\*\*/\*.java"  
myTree.builtBy "someTask"  
  
task copy(type: Copy) {  
 from myTree  
}

[**ConfigurableFileTree**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileTree.html) fileTree([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) baseDir, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). The closure will be used to configure the new file tree. The file tree is passed to the closure as its delegate. Example:

def myTree = fileTree('src') {  
 exclude '\*\*/.data/\*\*'  
 builtBy 'someTask'  
}  
  
task copy(type: Copy) {  
 from myTree  
}

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

[**ConfigurableFileTree**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileTree.html) fileTree([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) baseDir, [**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**ConfigurableFileTree**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileTree.html)> configureAction)

Creates a new ConfigurableFileTree using the given base directory. The given baseDir path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). The action will be used to configure the new file tree. Example:

def myTree = fileTree('src') {  
 exclude '\*\*/.data/\*\*'  
 builtBy 'someTask'  
}  
  
task copy(type: Copy) {  
 from myTree  
}

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

[**ConfigurableFileTree**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileTree.html) fileTree([**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), ?> args)

Creates a new ConfigurableFileTree using the provided map of arguments. The map will be applied as properties on the new file tree. Example:

def myTree = fileTree(dir:'src', excludes:['\*\*/ignore/\*\*', '\*\*/.data/\*\*'])  
  
task copy(type: Copy) {  
 from myTree  
}

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

[**ConfigurableFileCollection**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html) files([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) paths, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Creates a new ConfigurableFileCollection using the given paths. The paths are evaluated as per [Project.files(java.lang.Object[])](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object%5B%5D)). The file collection is configured using the given closure. The file collection is passed to the closure as its delegate. Example:

files "$buildDir/classes" {  
 builtBy 'compile'  
}

The returned file collection is lazy, so that the paths are evaluated only when the contents of the file collection are queried. The file collection is also live, so that it evaluates the above each time the contents of the collection is queried.

[**ConfigurableFileCollection**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html) files([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) paths, [**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**ConfigurableFileCollection**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html)> configureAction)

Creates a new ConfigurableFileCollection using the given paths. The paths are evaluated as per [Project.files(java.lang.Object[])](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:files(java.lang.Object%5B%5D)). The file collection is configured using the given action. Example:

files "$buildDir/classes" {  
 builtBy 'compile'  
}

The returned file collection is lazy, so that the paths are evaluated only when the contents of the file collection are queried. The file collection is also live, so that it evaluates the above each time the contents of the collection is queried.

[**ConfigurableFileCollection**](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html) files([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html)... paths)

Returns a [ConfigurableFileCollection](http://docs.google.com/javadoc/org/gradle/api/file/ConfigurableFileCollection.html) containing the given files. You can pass any of the following types to this method:

* A [CharSequence](http://download.oracle.com/javase/8/docs/api/java/lang/CharSequence.html), including [String](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) or [GString](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/GString.html). Interpreted relative to the project directory, as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). A string that starts with file: is treated as a file URL.
* A [File](http://download.oracle.com/javase/8/docs/api/java/io/File.html). Interpreted relative to the project directory, as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)).
* A [Path](http://download.oracle.com/javase/8/docs/api/java/nio/file/Path.html) as defined by [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)).
* A [URI](http://download.oracle.com/javase/8/docs/api/java/net/URI.html) or [URL](http://download.oracle.com/javase/8/docs/api/java/net/URL.html). The URL's path is interpreted as a file path. Only file: URLs are supported.
* A [Directory](http://docs.google.com/javadoc/org/gradle/api/file/Directory.html) or [RegularFile](http://docs.google.com/javadoc/org/gradle/api/file/RegularFile.html).
* A [Collection](http://download.oracle.com/javase/8/docs/api/java/util/Collection.html), [Iterable](http://download.oracle.com/javase/8/docs/api/java/lang/Iterable.html), or an array that contains objects of any supported type. The elements of the collection are recursively converted to files.
* A [FileCollection](http://docs.google.com/javadoc/org/gradle/api/file/FileCollection.html). The contents of the collection are included in the returned collection.
* A [Provider](http://docs.google.com/javadoc/org/gradle/api/provider/Provider.html) of any supported type. The provider's value is recursively converted to files. If the provider represents an output of a task, that task is executed if the file collection is used as an input to another task.
* A [Callable](http://download.oracle.com/javase/8/docs/api/java/util/concurrent/Callable.html) that returns any supported type. The return value of the call() method is recursively converted to files. A null return value is treated as an empty collection.
* A [Closure](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) that returns any of the types listed here. The return value of the closure is recursively converted to files. A null return value is treated as an empty collection.
* A [Task](http://docs.google.com/dsl/org.gradle.api.Task.html). Converted to the task's output files. The task is executed if the file collection is used as an input to another task.
* A [TaskOutputs](http://docs.google.com/javadoc/org/gradle/api/tasks/TaskOutputs.html). Converted to the output files the related task. The task is executed if the file collection is used as an input to another task.
* Anything else is treated as a failure.

The returned file collection is lazy, so that the paths are evaluated only when the contents of the file collection are queried. The file collection is also live, so that it evaluates the above each time the contents of the collection is queried.

The returned file collection maintains the iteration order of the supplied paths.

The returned file collection maintains the details of the tasks that produce the files, so that these tasks are executed if this file collection is used as an input to some task.

This method can also be used to create an empty collection, which can later be mutated to add elements.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) findProject([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path)

Locates a project by path. If the path is relative, it is interpreted relative to this project.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) findProperty([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) propertyName)

Returns the value of the given property or null if not found. This method locates a property as follows:

1. If this project object has a property with the given name, return the value of the property.
2. If this project has an extension with the given name, return the extension.
3. If this project's convention object has a property with the given name, return the value of the property.
4. If this project has an extra property with the given name, return the value of the property.
5. If this project has a task with the given name, return the task.
6. Search up through this project's ancestor projects for a convention property or extra property with the given name.
7. If not found, null value is returned.

[**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html), [**Set**](http://download.oracle.com/javase/8/docs/api/java/util/Set.html)<[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html)>> getAllTasks(boolean recursive)

Returns a map of the tasks contained in this project, and optionally its subprojects.

[**Set**](http://download.oracle.com/javase/8/docs/api/java/util/Set.html)<[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html)> getTasksByName([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name, boolean recursive)

Returns the set of tasks with the given name contained in this project, and optionally its subprojects. *NOTE:* This is an expensive operation since it requires all projects to be configured.

boolean hasProperty([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) propertyName)

Determines if this project has the given property. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for details of the properties which are available for a project.

[**ExecResult**](http://docs.google.com/javadoc/org/gradle/process/ExecResult.html) javaexec([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Executes a Java main class. The closure configures a [JavaExecSpec](http://docs.google.com/javadoc/org/gradle/process/JavaExecSpec.html).

[**ExecResult**](http://docs.google.com/javadoc/org/gradle/process/ExecResult.html) javaexec([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**JavaExecSpec**](http://docs.google.com/javadoc/org/gradle/process/JavaExecSpec.html)> action)

Executes an external Java process.

The given action configures a [JavaExecSpec](http://docs.google.com/javadoc/org/gradle/process/JavaExecSpec.html), which is used to launch the process. This method blocks until the process terminates, with its result being returned.

[**File**](http://download.oracle.com/javase/8/docs/api/java/io/File.html) mkdir([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) path)

Creates a directory and returns a file pointing to it.

void normalization([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**InputNormalizationHandler**](http://docs.google.com/dsl/org.gradle.normalization.InputNormalizationHandler.html)> configuration)

Note: This method is [incubating](http://docs.google.com/userguide/feature_lifecycle.html) and may change in a future version of Gradle.

Configures input normalization.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) project([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path)

Locates a project by path. If the path is relative, it is interpreted relative to this project.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) project([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Locates a project by path and configures it using the given closure. If the path is relative, it is interpreted relative to this project. The target project is passed to the closure as the closure's delegate.

[**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html) project([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path, [**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> configureAction)

Locates a project by path and configures it using the given action. If the path is relative, it is interpreted relative to this project.

[**PropertyState**](http://docs.google.com/javadoc/org/gradle/api/provider/PropertyState.html)<T> property([**Class**](http://download.oracle.com/javase/8/docs/api/java/lang/Class.html)<T> clazz)

Note: This method is [deprecated](http://docs.google.com/userguide/feature_lifecycle.html) and will be removed in the next major version of Gradle.

Note: This method is [incubating](http://docs.google.com/userguide/feature_lifecycle.html) and may change in a future version of Gradle.

Creates a PropertyState implementation based on the provided class.

[**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) property([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) propertyName)

Returns the value of the given property. This method locates a property as follows:

1. If this project object has a property with the given name, return the value of the property.
2. If this project has an extension with the given name, return the extension.
3. If this project's convention object has a property with the given name, return the value of the property.
4. If this project has an extra property with the given name, return the value of the property.
5. If this project has a task with the given name, return the task.
6. Search up through this project's ancestor projects for a convention property or extra property with the given name.
7. If not found, a [MissingPropertyException](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/MissingPropertyException.html) is thrown.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) relativePath([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) path)

Returns the relative path from the project directory to the given path. The given path object is (logically) resolved as described for [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)), from which a relative path is calculated.

[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) relativeProjectPath([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) path)

Converts a name to a project path relative to this project.

void setProperty([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name, [**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) value)

Sets a property of this project. This method searches for a property with the given name in the following locations, and sets the property on the first location where it finds the property.

1. The project object itself. For example, the rootDir project property.
2. The project's [Convention](http://docs.google.com/javadoc/org/gradle/api/plugins/Convention.html) object. For example, the srcRootName java plugin property.
3. The project's extra properties.

If the property is not found, a [MissingPropertyException](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/MissingPropertyException.html) is thrown.

void subprojects([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Project**](http://docs.google.com/dsl/org.gradle.api.Project.html)> action)

Configures the sub-projects of this project

This method executes the given [Action](http://docs.google.com/javadoc/org/gradle/api/Action.html) against the sub-projects of this project.

[**WorkResult**](http://docs.google.com/javadoc/org/gradle/api/tasks/WorkResult.html) sync([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**CopySpec**](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html)> action)

Synchronizes the contents of a destination directory with some source directories and files. The given action is used to configure a [CopySpec](http://docs.google.com/javadoc/org/gradle/api/file/CopySpec.html), which is then used to synchronize the files.

This method is like the [Project.copy(org.gradle.api.Action)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:copy(org.gradle.api.Action)) task, except the destination directory will only contain the files copied. All files that exist in the destination directory will be deleted before copying files, unless a preserve option is specified.

Example:

project.sync {  
 from 'my/shared/dependencyDir'  
 into 'build/deps/compile'  
}

Note that you can preserve output that already exists in the destination directory:

project.sync {  
 from 'source'  
 into 'dest'  
 preserve {  
 include 'extraDir/\*\*'  
 include 'dir1/\*\*'  
 exclude 'dir1/extra.txt'  
 }  
}

[**FileTree**](http://docs.google.com/javadoc/org/gradle/api/file/FileTree.html) tarTree([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) tarPath)

Creates a new FileTree which contains the contents of the given TAR file. The given tarPath path can be:

* an instance of [Resource](http://docs.google.com/javadoc/org/gradle/api/resources/Resource.html)
* any other object is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object))

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

Unless custom implementation of resources is passed, the tar tree attempts to guess the compression based on the file extension.

You can combine this method with the [Project.copy(groovy.lang.Closure)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:copy(groovy.lang.Closure)) method to untar a TAR file:

task untar(type: Copy) {  
 from tarTree('someCompressedTar.gzip')  
  
 //tar tree attempts to guess the compression based on the file extension  
 //however if you must specify the compression explicitly you can:  
 from tarTree(resources.gzip('someTar.ext'))  
  
 //in case you work with unconventionally compressed tars  
 //you can provide your own implementation of a ReadableResource:  
 //from tarTree(yourOwnResource as ReadableResource)  
  
 into 'dest'  
}

[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html) task([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name)

Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Calling this method is equivalent to calling [Project.task(java.util.Map, java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String)) with an empty options map.

After the task is added to the project, it is made available as a property of the project, so that you can reference the task by name in your build file. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

If a task with the given name already exists in this project, an exception is thrown.

[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html) task([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given closure is executed to configure the task.

After the task is added to the project, it is made available as a property of the project, so that you can reference the task by name in your build file. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html) task([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name, [**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html)> configureAction)

Note: This method is [incubating](http://docs.google.com/userguide/feature_lifecycle.html) and may change in a future version of Gradle.

Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given action is executed to configure the task.

After the task is added to the project, it is made available as a property of the project, so that you can reference the task by name in your build file. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html) task([**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), ?> args, [**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name)

Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. A map of creation options can be passed to this method to control how the task is created. The following options are available:

| Option | Description | Default Value |
| --- | --- | --- |
| type | The class of the task to create. | [DefaultTask](http://docs.google.com/dsl/org.gradle.api.DefaultTask.html) |
| overwrite | Replace an existing task? | false |
| dependsOn | A task name or set of task names which this task depends on | [] |
| action | A closure or [Action](http://docs.google.com/javadoc/org/gradle/api/Action.html) to add to the task. | null |
| description | A description of the task. | null |
| group | A task group which this task belongs to. | null |

After the task is added to the project, it is made available as a property of the project, so that you can reference the task by name in your build file. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

If a task with the given name already exists in this project and the override option is not set to true, an exception is thrown.

[**Task**](http://docs.google.com/dsl/org.gradle.api.Task.html) task([**Map**](http://download.oracle.com/javase/8/docs/api/java/util/Map.html)<[**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html), ?> args, [**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) name, [**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Creates a [Task](http://docs.google.com/dsl/org.gradle.api.Task.html) with the given name and adds it to this project. Before the task is returned, the given closure is executed to configure the task. A map of creation options can be passed to this method to control how the task is created. See [Project.task(java.util.Map, java.lang.String)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:task(java.util.Map,%20java.lang.String)) for the available options.

After the task is added to the project, it is made available as a property of the project, so that you can reference the task by name in your build file. See [here](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project.properties) for more details

If a task with the given name already exists in this project and the override option is not set to true, an exception is thrown.

[**URI**](http://download.oracle.com/javase/8/docs/api/java/net/URI.html) uri([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) path)

Resolves a file path to a URI, relative to the project directory of this project. Evaluates the provided path object as described for [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)), with the exception that any URI scheme is supported, not just 'file:' URIs.

[**FileTree**](http://docs.google.com/javadoc/org/gradle/api/file/FileTree.html) zipTree([**Object**](http://download.oracle.com/javase/8/docs/api/java/lang/Object.html) zipPath)

Creates a new FileTree which contains the contents of the given ZIP file. The given zipPath path is evaluated as per [Project.file(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:file(java.lang.Object)). You can combine this method with the [Project.copy(groovy.lang.Closure)](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:copy(groovy.lang.Closure)) method to unzip a ZIP file.

The returned file tree is lazy, so that it scans for files only when the contents of the file tree are queried. The file tree is also live, so that it scans for files each time the contents of the file tree are queried.

void appDirName([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) appDirName)

Allows changing the application directory. Default is "src/main/application".

[**EarPluginConvention**](http://docs.google.com/dsl/org.gradle.plugins.ear.EarPluginConvention.html) deploymentDescriptor([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**DeploymentDescriptor**](http://docs.google.com/javadoc/org/gradle/plugins/ear/descriptor/DeploymentDescriptor.html)> configureAction)

Configures the deployment descriptor for this EAR archive.

The given action is executed to configure the deployment descriptor.

void libDirName([**String**](http://download.oracle.com/javase/8/docs/api/java/lang/String.html) libDirName)

Allows changing the library directory in the EAR file. Default is "lib".

[**Manifest**](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html) manifest()

Creates a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html).

[**Manifest**](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html) manifest([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Creates and configures a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html). The given closure configures the new manifest instance before it is returned.

[**Manifest**](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html) manifest([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**Manifest**](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html)> action)

Creates and configures a new instance of a [Manifest](http://docs.google.com/javadoc/org/gradle/api/java/archives/Manifest.html).

[**MavenPom**](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html) pom()

Creates a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html).

[**MavenPom**](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html) pom([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) configureClosure)

Creates and configures a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html). The given closure is executed to configure the new POM instance.

[**MavenPom**](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html) pom([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**MavenPom**](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html)> configureAction)

Creates and configures a new [MavenPom](http://docs.google.com/javadoc/org/gradle/api/artifacts/maven/MavenPom.html). The given action is executed to configure the new POM instance.

[**OsgiManifest**](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html) osgiManifest()

Creates a new instance of [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html). The returned object is preconfigured with:

version: project.version  
name: project.archivesBaseName  
symbolicName: project.group + "." + project.archivesBaseName (see below for exceptions to this rule)

The symbolic name is usually the group + "." + archivesBaseName, with the following exceptions

* if group has only one section (no dots) and archivesBaseName is not null then the first package name with classes is returned. eg. commons-logging:commons-logging -> org.apache.commons.logging
* if archivesBaseName is equal to last section of group then group is returned. eg. org.gradle:gradle -> org.gradle
* if archivesBaseName starts with last section of group that portion is removed. eg. org.gradle:gradle-core -> org.gradle.core

[**OsgiManifest**](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html) osgiManifest([**Closure**](http://docs.groovy-lang.org/2.4.15/html/gapi/groovy/lang/Closure.html) closure)

Creates and configures a new instance of an [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html) . The closure configures the new manifest instance before it is returned.

[**OsgiManifest**](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html) osgiManifest([**Action**](http://docs.google.com/javadoc/org/gradle/api/Action.html)<? super [**OsgiManifest**](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html)> action)

Creates and configures a new instance of an [OsgiManifest](http://docs.google.com/javadoc/org/gradle/api/plugins/osgi/OsgiManifest.html). The action configures the new manifest instance before it is returned.

Script block details

allprojects { }

Configures this project and each of its sub-projects.

This method executes the given closure against this project and its sub-projects. The target [Project](http://docs.google.com/dsl/org.gradle.api.Project.html) is passed to the closure as the closure's delegate.

Delegates to:Each [Project](http://docs.google.com/dsl/org.gradle.api.Project.html) in [allprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:allprojects)

ant { }

Executes the given closure against the AntBuilder for this project. You can use this in your build file to execute ant tasks. The AntBuild is passed to the closure as the closure's delegate. See example in javadoc for [Project.getAnt()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:ant)

Delegates to:[AntBuilder](http://docs.google.com/javadoc/org/gradle/api/AntBuilder.html) from [ant](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:ant)

artifacts { }

Configures the published artifacts for this project.

This method executes the given closure against the [ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) for this project. The [ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) is passed to the closure as the closure's delegate.

Example:

configurations {  
 //declaring new configuration that will be used to associate with artifacts  
 schema  
}  
  
task schemaJar(type: Jar) {  
 //some imaginary task that creates a jar artifact with the schema  
}  
  
//associating the task that produces the artifact with the configuration  
artifacts {  
 //configuration name and the task:  
 schema schemaJar  
}

Delegates to:[ArtifactHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.ArtifactHandler.html) from [artifacts](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:artifacts)

buildscript { }

Configures the build script classpath for this project.

The given closure is executed against this project's [ScriptHandler](http://docs.google.com/javadoc/org/gradle/api/initialization/dsl/ScriptHandler.html). The [ScriptHandler](http://docs.google.com/javadoc/org/gradle/api/initialization/dsl/ScriptHandler.html) is passed to the closure as the closure's delegate.

Delegates to:[ScriptHandler](http://docs.google.com/javadoc/org/gradle/api/initialization/dsl/ScriptHandler.html) from [buildscript](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:buildscript)

configurations { }

Configures the dependency configurations for this project.

This method executes the given closure against the [ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html) for this project. The [ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html) is passed to the closure as the closure's delegate.

Examples:

See docs for [ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html)

Delegates to:[ConfigurationContainer](http://docs.google.com/dsl/org.gradle.api.artifacts.ConfigurationContainer.html) from [configurations](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:configurations)

dependencies { }

Configures the dependencies for this project.

This method executes the given closure against the [DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html) for this project. The [DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html) is passed to the closure as the closure's delegate.

Examples:

See docs for [DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html)

Delegates to:[DependencyHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.DependencyHandler.html) from [dependencies](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:dependencies)

repositories { }

Configures the repositories for this project.

This method executes the given closure against the [RepositoryHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.RepositoryHandler.html) for this project. The [RepositoryHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.RepositoryHandler.html) is passed to the closure as the closure's delegate.

Delegates to:[RepositoryHandler](http://docs.google.com/dsl/org.gradle.api.artifacts.dsl.RepositoryHandler.html) from [repositories](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:repositories)

subprojects { }

Configures the sub-projects of this project.

This method executes the given closure against each of the sub-projects of this project. The target [Project](http://docs.google.com/dsl/org.gradle.api.Project.html) is passed to the closure as the closure's delegate.

Delegates to:Each [Project](http://docs.google.com/dsl/org.gradle.api.Project.html) in [subprojects](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:subprojects)

announce { }

Configures the [AnnouncePluginExtension](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) added by the announce plugin.

Delegates to:[AnnouncePluginExtension](http://docs.google.com/dsl/org.gradle.api.plugins.announce.AnnouncePluginExtension.html) from [announce](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:announce)

checkstyle { }

Configures the [CheckstyleExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) added by the checkstyle plugin.

Delegates to:[CheckstyleExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CheckstyleExtension.html) from [checkstyle](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:checkstyle)

codenarc { }

Configures the [CodeNarcExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) added by the codenarc plugin.

Delegates to:[CodeNarcExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarcExtension.html) from [codenarc](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:codenarc)

distributions { }

Configures the [DistributionContainer](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) added by the distribution plugin.

Delegates to:[DistributionContainer](http://docs.google.com/dsl/org.gradle.api.distribution.DistributionContainer.html) from [distributions](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:distributions)

deploymentDescriptor { }

Configures the deployment descriptor for this EAR archive.

The given closure is executed to configure the deployment descriptor. The [DeploymentDescriptor](http://docs.google.com/javadoc/org/gradle/plugins/ear/descriptor/DeploymentDescriptor.html) is passed to the closure as its delegate.

Delegates to:[DeploymentDescriptor](http://docs.google.com/javadoc/org/gradle/plugins/ear/descriptor/DeploymentDescriptor.html) from [deploymentDescriptor](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:deploymentDescriptor)

eclipse { }

Configures the [EclipseModel](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) added by the eclipse plugin.

Delegates to:[EclipseModel](http://docs.google.com/dsl/org.gradle.plugins.ide.eclipse.model.EclipseModel.html) from [eclipse](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:eclipse)

findbugs { }

Configures the [FindBugsExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) added by the findbugs plugin.

Delegates to:[FindBugsExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugsExtension.html) from [findbugs](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:findbugs)

idea { }

Configures the [IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html) added by the idea plugin.

Delegates to:[IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html) from [idea](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:idea)

jacoco { }

Configures the [JacocoPluginExtension](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) added by the jacoco plugin.

Delegates to:[JacocoPluginExtension](http://docs.google.com/dsl/org.gradle.testing.jacoco.plugins.JacocoPluginExtension.html) from [jacoco](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jacoco)

reporting { }

Configures the [ReportingExtension](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) added by the java plugin.

Delegates to:[ReportingExtension](http://docs.google.com/dsl/org.gradle.api.reporting.ReportingExtension.html) from [reporting](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:reporting)

sourceSets { }

Configures the source sets of this project.

The given closure is executed to configure the [SourceSetContainer](http://docs.google.com/javadoc/org/gradle/api/tasks/SourceSetContainer.html). The [SourceSetContainer](http://docs.google.com/javadoc/org/gradle/api/tasks/SourceSetContainer.html) is passed to the closure as its delegate.

See the example below how [SourceSet](http://docs.google.com/dsl/org.gradle.api.tasks.SourceSet.html) 'main' is accessed and how the [SourceDirectorySet](http://docs.google.com/dsl/org.gradle.api.file.SourceDirectorySet.html) 'java' is configured to exclude some package from compilation.

apply plugin: 'java'  
  
sourceSets {  
 main {  
 java {  
 exclude 'some/unwanted/package/\*\*'  
 }  
 }  
}

Delegates to:[SourceSetContainer](http://docs.google.com/javadoc/org/gradle/api/tasks/SourceSetContainer.html) from [sourceSets](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:sourceSets)

jdepend { }

Configures the [JDependExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) added by the jdepend plugin.

Delegates to:[JDependExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDependExtension.html) from [jdepend](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:jdepend)

pmd { }

Configures the [PmdExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) added by the pmd plugin.

Delegates to:[PmdExtension](http://docs.google.com/dsl/org.gradle.api.plugins.quality.PmdExtension.html) from [pmd](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:pmd)

publishing { }

Configures the [PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) added by the publishing plugin.

Delegates to:[PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html) from [publishing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:publishing)

signing { }

Configures the [SigningExtension](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) added by the signing plugin.

Delegates to:[SigningExtension](http://docs.google.com/dsl/org.gradle.plugins.signing.SigningExtension.html) from [signing](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:signing)

visualStudio { }

Configures the [VisualStudioRootExtension](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) added by the visual-studio plugin.

Delegates to:[VisualStudioRootExtension](http://docs.google.com/dsl/org.gradle.ide.visualstudio.VisualStudioRootExtension.html) from [visualStudio](http://docs.google.com/org.gradle.api.Project.html#org.gradle.api.Project:visualStudio)

**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/)