# <https://www.sghill.net/5-minute-intro-to-gradle.html>

# Learn Gradle in x minutes

Gradle is build automation evolved. Gradle can automate the building, testing, publishing, deployment and more of software packages or other types of projects such as generated static websites, generated documentation or indeed anything else.

Gradle combines the power and flexibility of Ant with the dependency management and conventions of Maven into a more effective way to build. Powered by a Groovy DSL and packed with innovation, Gradle provides a declarative way to describe all kinds of builds through sensible defaults.

## Installation

Of course, you can download and install Gradle from <http://www.gradle.org/downloads>.

And set the path > sudo ~/.bashrc

## Hello

Create a file called build.gradle. This is a build script.

task hello {

doLast {

println 'Hello world!'

}

}

Execute the build script with gradle -q hello:

$ gradle -q hello

Hello world!

## Concept basics

There are some basic concepts that you shold know.

### Gradle scripts are configuration scripts

As the script executes, it configures an object of a particular type. For example, as a build script executes, it configures an object of type Project. There is a one-to-one relationship between a Project and a build.gradle file.

A project is essentially a collection of Task objects. Each task performs some basic piece of work, such as compiling classes, or running unit tests

The following table shows the delegate for each type of Gradle script.

| Type of script | Delegates to instance of |
| --- | --- |
| Build script | Project |
| Init script | Gradle |
| Settings script | Settings |

The properties and methods of the delegate object are available for you to use in the script.

### Each Gradle script implements the Script interface.

This interface defines a number of properties and methods which you can use in the script.

A build script is made up of zero or more statements and script blocks. Statements can include method calls, property assignments, and local variable definitions.

## Command-Line

USAGE: gradle [option...] [task...]

There are some help tasks you should know, which can help you to write build script and find problem.

### tasks

See all tasks runnable from a project by typing:

$ gradle <project-path>:tasks

And you can see all tasks you defined and help tasks as well.

For the root project, you can just type gradle tasks

### projects

See the sub-projects of a project by typing:

$ gradle <project-path>:projects

### properties

See the properties of a project by typing:

$ gradle <project-path>:properties

It displays a log of properties like:

* project
* rootProject
* allprojects
* subprojects
* tasks
* buildDir
* projectDir

### dependencies

See the dependencies declared of a project by typing:

$ gradle <project-path>:dependencies

## Build Script Basics

### Projects and tasks

Everything in Gradle sits on top of two basic concepts: projects and tasks. Every Gradle build is made up of one or more projects. Each project is made up of one or more tasks. A task represents some atomic piece of work which a build performs.

### Task definition

build.gradle

task hello {

doLast {

println 'Hello world!'

}

}

There is another easy way to define a task:

task hello << {

println 'Hello world!'

}

### Using Groovy in tasks

build.gradle

task hello << {

String name = 'jeoygin'

println "Hello " + name.toUpperCase()

}

Output of gradle -q hello

$ gradle -q hello

Hello JEOYGIN

or more complicated.

task count << {

4.times {print "${it + 1} "}

}

Output of gradle -q count

$ gradle -q count

1 2 3 4

The value of variable can be read in a string by $var or ${var}.

### Dependency

build.gradle

task taskX(dependsOn: 'taskY') << {

println 'taskX'

}

task taskY << {

println 'taskY'

}

Output of gradle -q taskX

$ gradle -q taskX

taskY

taskX

### Dynamic tasks

build.gradle

4.times { counter ->

task "task$counter" << {

println "I'm task number $counter"

}

}

Part of output of gradle -q tasks

$ gradle -q tasks

Other tasks

-----------

task0

task1

task2

task3

Output of gradle -q task0

$ gradle -q task0

I'm task number 1

### Manipulating existing tasks

build.gradle

4.times { counter ->

task "task$counter" << {

println "I'm task number $counter"

}

}

task0.dependsOn task2, task3

Output of gradl -q task0

$ gradle -q task0

I'm task number 2

I'm task number 3

I'm task number 0

Or add behavior to an existing task.

build.gradle

task hello << {

println 'Hello Earth'

}

hello.doFirst {

println 'Hello Venus'

}

hello.doLast {

println 'Hello Mars'

}

hello << {

println 'Hello Jupiter'

}

Output of gradle -q hello

$ gradle -q hello

Hello Venus

Hello Earth

Hello Mars

Hello Jupiter

### Extra task properties

You can add your own properties to a task. To add a property named myProperty, set ext.myProperty to an initial value.

build.gradle

task test {

ext.extraProperty = "extra"

}

task printExtraProperties << {

println test.extraProperty

}

Output of gradle -q printExtraProperties

$ gradle -q printExtraProperties

extra

### Using methods

build.gradle

task hello {

sayHello('Jeoygin')

}

String sayHello(String name) {

println "Hello $name"

}

Output of gradle -q hello

$ gradle -q hello

Hello Jeoygin

### Default tasks

build.gradle

defaultTasks 'clean', 'run'

task clean << {

println 'Default Cleaning!'

}

task run << {

println 'Default Running!'

}

task other << {

println "I'm not a default task!"

}

Output of gradle -q

$ gradle -q

Default Cleaning!

Default Running!

### Configure by DAG

build.gradle

task distribution << {

println "Build the project with version=$version"

}

task release(dependsOn: 'distribution') << {

println 'Release the project'

}

gradle.taskGraph.whenReady {taskGraph ->

if (taskGraph.hasTask(release)) {

version = '1.0'

} else {

version = '1.0-SNAPSHOT'

}

}

Output of gradle -q distribution

$ gradle -q distribution

Build the project with version=1.0-SNAPSHOT

Output of gradle -q release

$ gradle -q release

Build the project with version=1.0

Release the project

## Reference

1. [Gradle User Guild](http://www.gradle.org/docs/current/userguide/userguide.html)
2. [Gradle DSL Reference](http://www.gradle.org/docs/current/dsl/index.html)
3. [Gradle Javadoc](http://www.gradle.org/docs/current/javadoc/)