Gradle is a project build and automation tool for java based applications; something like ivy, ant and maven. Build tools help us to reduce our development, test time, and hence increase productivity.

### What is Gradle:

Gradle is an opensource build and automation tool for java based projects. Using Gradle, we can reduce project development time and increase productivity.

Gradle is a multi-language, multi-platform, multi-project and multi-channel build and automation software.

**Drawbacks of Ant:**

The following are main drawbacks of using ant build tools in our project.

1. We need to write ant build scripts using XML. If we want to automate a complex project, then we need to write a lot of logic in XML files.
2. When we execute complex and large project’s ant build, it produces a lot of verbose at console.
3. There is no built-in ant project structure. Since we can use any build structure for our projects, new developers find it hard to understand the project struct and build scripts.
4. It is very tough to write some complex logic using if-then-else statements.
5. We need to maintain all jars with required version in version control. There is no support for dependency management.

**Advantages of Maven:**

1. Maven is an expressive, declarative, and maintainable build tool.
2. All maven projects follow pre-defined structure. Even new developers find it easy to understand the project structure and start development quickly.
3. We don’t need to maintain all jars with required version in version control. Maven will download all required jars at build time. Maven support for dependency management is one of the best advantage it has over ant.
4. Maven gives us very modularized project structure

**Drawbacks of Maven:**

1. Maven follows some pre-defined build and automation lifecycle. Sometimes it may not fit to our project needs.
2. Even though we can write custom maven life cycle methods, but it’s a bit tough and verbose.
3. Maven build scripts are a bit tough to maintain for very complex projects.

**Gradle advantages**

Gradle will provide the following advantages compared to ant and maven. That’s why all new projects are using Gradle as build tool.

1. Like Maven, Gradle is also an expressive, declarative, and maintainable build Tool.
2. Just like maven, Gradle also supports dependency management.
3. Gradle provides very scalable and high-performance builds.
4. Gradle provides standard project layout and lifecycle, but it’s full flexibility. We have the option to fully configure the defaults. This is where it’s better than maven.
5. It’s very easy to use gradle tool and implement custom logic in our project.
6. Gradle supports the project structure that consists of more than one project to build deliverable.
7. It is very easy to integrate existing ant/maven project with Gradle.
8. It is very easy to migrate from existing ant/maven project to Gradle.

Gradle uses it’s own DSL(Domain Specific Language) Groovy-based scripts to write build scripts. Unlike Ant and Maven, it does not use complex XML build scripts. Gradleis developed using Java and configuration elements are developed using Groovy. It uses Groovy to write build scripts.

First and foremost thing we should know is Gradle default build script file name. Like Ant’s default build script name is build.xml and Maven default build script name is pom.xml, Gradle default build script name is build.gradle".

When we run “gradle” command, it searches for this default file available in the current working directory. If it finds, it executes that build script. Otherwise, displays some useful default help message.

**Gradle Build:**

A project is either something we build (ex: a jar file) or do (deploy our application to production environment) A project consist of one or more tasks.

A task is atomic unit work which is performed our build(compiling our project or running tests)

Or

Each project is made up of one or more tasks, a task represents some atomic piece of work which build performs, this might be compiling some classes, creating a JAR file, generating Javadoc or publishing some archives to a repository.

Every gradle build contains one or more projects.

Build---🡪 Projects ---🡪 tasks = one to many, means one build and many projects and tasks.

**Gradle Configuration Files**

**build.gradle:**

Thegradle build scrips (build.gradle) specifies a project and its tasks.

**Gradle.properties:**

The gradle properties file (gradle.properties) is used to configure the properties of the build.

**Settings.gradle:**

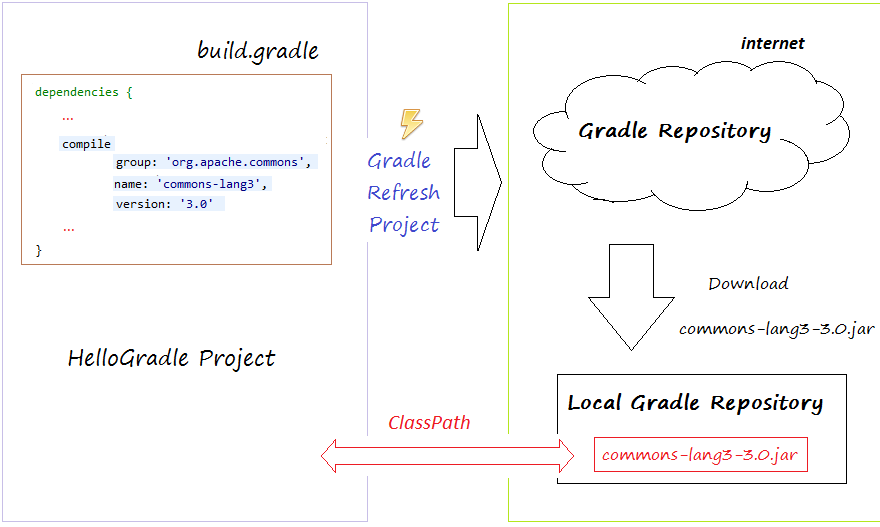
The gradle settings file(settings.gradle) is optional in a build which has only one project.

If our gradle build has more than one project, it is mandatory because it describes which projects participate to our build, every multi project build must have a settings file in the root project of the project hierarchy.

**.gradle:**

. gradle is local repository it contains all the dependent jars and located user home directory.

**Gradle dependencies process:**



**Interview Questions and Answers:**

**What is Gradleframework**

Gradle is an open source build automation system that builds based on the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language (DSL) instead of the XML form used by Apache Maven for declaring the project configuration.

**Advantages of using Gradle**

Gradle combines both Ant and Maven, taking the best from both of these frameworks; Flexibility from Ant tool and convention over configuration, dependency management and plugins from Maven.

Gradle provides support for multi-project builds.

It allows to use existing Maven/Ivy repositories.

Free and open source.

Scalable nature.

**What is Gradlewrapper**

A wrapper is a batch script and it is one of the ways to perform Gradle build. When executed the first time, it automatically downloads Gradle and then initiate the build.

It helps to setup Gradle workspace quickly for first-time users (Zero installation) and also ensure all the developers use the same version of Gradle.

**Why Gradle is preferred over other build tools**

Gradle build script is written using groovy API which has the syntax similar to Java so it is easy to understand.

Gradle supports ant tasks, ivy and Maven repositories for dependency management. It also has a maven Pom.xml converter to Gradle build script.

It is open source.

provides strong support for multi project builds.

supports build cache.

**What is the Gradle build script file name**

build.gradle.

**Gradel /default generator package file:**

As “jar” is default assemble value, we don’t need to apply jar plugin in gradle’s build script file. However, to create WAR or EAR we need to apply the respective plugins as shown below:

**This gradle element definition;**

**dependencies {**

**compile group:'commons-collections',name:'commons-collections',version:'3.2'**

**}**

is similar to the following Maven definition

**<dependencies>**

**<dependency>**

**<groupId>commons-collections</groupId>**

**<artifactId>commons-collections</artifactId>**

**<version>3.2</version>**

**</dependency>**

**<dependencies>**

That means Gradle’s “group” element is similar to Maven’s “groupId” element.

Gradle’s “name” element is similar to Maven’s “artifactId” element and Gradle’s “version” element is similar to Maven’s “version” element.

**Dradle dependencies location:**

C:\Users\MekalaNagendra\.gradle\caches\modules-2\files-2.1

**maven dependencies location:**

C:\Users\[Windos-UserName]\.m2\repositor

**Maven vs Gradle Build Script Major Elements**

If we observe both maven’s pom.xml and gradle’s build.gradle file, we can find the following things.

|  |  |
| --- | --- |
| MAVEN ELEMENTS | GRADLE ELEMENTS |
| <groupId> | Gradle’s “group” Element |
| <artifactId> | Gradle’s “baseName” Element |
| <version> | Gradle’s “version” Element |

* Like Maven, we can integrate existing ANT tasks in Gradle and use them in our projects.
* Like Maven, we can create our own project specific Gradle tasks and use them in our projects.
* We can even Integrate Maven tasks with Gradle build tool.

Maven vs Gradle commands

The following table lists all important maven and gradle commands.

|  |  |
| --- | --- |
| MAVEN COMMAND | GRADLE COMMAND |
| mvn package | gradle assemble |
| mvn test | gradle test |
| mvn clean | gradle clean |
| mvn –help | gradle –help |
| mvn install | gradle install |
| mvn –version | gradle –version |

Why gradle

**Fast:**

gradle completes tasks quickly by reusing outputs from previous executions, processing only the input changes and parallel processing.

**Highly customizable:**

gradle is supported multi language, so we can customize easily.

Interview questions and answers site:

[https://www.javapedia.net/Gradle-interview-questions#qanda1664](https://www.javapedia.net/Gradle-interview-questions" \l "qanda1664)

<https://proandroiddev.com/understanding-gradle-the-build-lifecycle-5118c1da613f>