Dependency management: repositories

Small programs that you write when learning a language may not use any external libraries at all. When you need some functionality, you will find it in the standard library or create it yourself.

However, it is quite difficult to develop a real application that doesn't use any libraries because they save tons of your time and provide solutions tested by millions of people around the world.

How do you add dependencies?
In Gradle terminology, all external libraries are called **dependencies**.

As a rule, they are packaged in JAR files. Gradle can automatically download them and add them to the project. It saves a lot of time and solves possible conflicts between versions of libraries.

Where do we get these dependencies, and how do we add them to a project?

To use a class, you need to have it locally, and your JVM must know that you have it. If you want to do it manually, you need to find and download such jars on your own and add them to the classpath of your project. Sounds quite tedious, right?

Fortunately, dependency management is one of the key features of Gradle. You don’t even need a plugin for it. To add an external library to a project, you need to do exactly two steps:

1. **Define a repository** where to search for dependencies.
2. **Define a dependency** that you want to include in your project.

Repositories and where to find them

Repositories are just places where libraries are stored.
Any project can use zero or more repositories.

There are different possible formats of repositories:

- a Maven compatible repository (e.g.: Maven Central, Google)
- an Ivy compatible repository;
- local (flat) directories.

It's possible to host repositories like **Maven** locally in your organization, but that is out of the scope of this tutorial. We will only consider public online versions of them.

Gradle has four aliases that you can use when adding Maven compatible repositories to the project.

- mavenCentral(): fetches dependencies from the Maven Central Repository.
- mavenLocal(): fetches dependencies from the local Maven repository.
- google(): fetches dependencies from Google Maven repository.

Repository definition

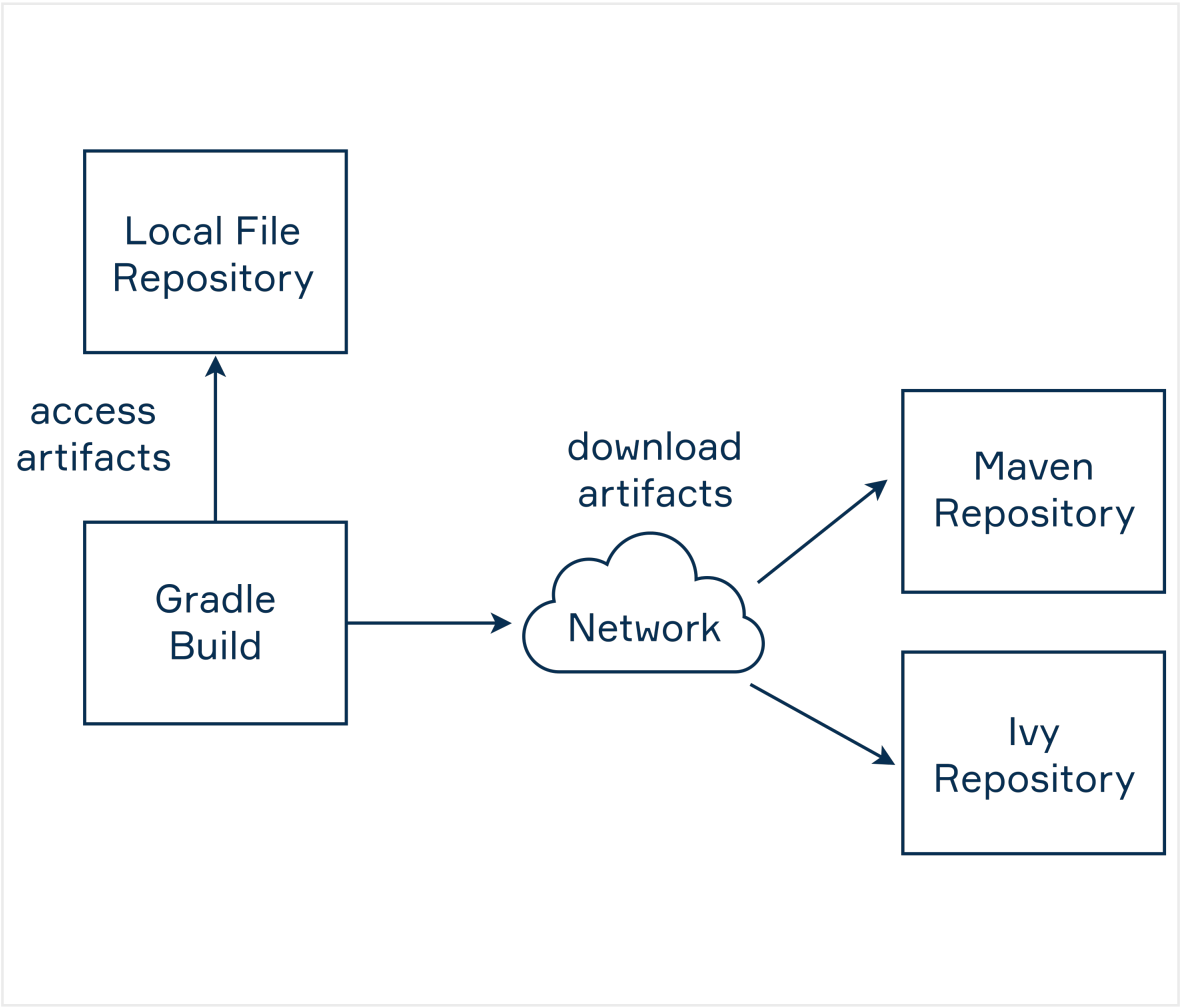
Defining a repository in Gradle is a piece of cake. Just add this to your build.gradle file:

```
repositories {
    mavenCentral()
}
```

Also, you can just download the jars you need and place them into some directory on your computer, commonly in the *libs* folder of your project. This comes in handy when the jars you need are not available in public repositories.

```
repositories {
    flatDir {
        dirs 'libs'
    }
}
```

The following picture demonstrates how to add dependencies from different repositories using Gradle.


Local File Repository
access artifacts
Gradle Build
download artifacts
Network
Maven Repository
Ivy Repository