What is Gradle in Android Studio?

Ask Question

1195

391

Gradle is bit confusing to me and also for new Android developer. Can anyone explain what gradle in Android Studio is and what its purpose is? Why is Gradle included in Android Studio?

android-studio gradle android-gradle build-automation

edited Feb 27 at 14:40

Zoe
14.4k 8 56 87

asked May 25 '13 at 22:53

androidcodehunter
9,339 15 40 63

4 See gradle.org – Eng.Fouad May 25 '13 at 22:59

12 To answer *why* you would want a build tool such as Gradle, check out this post from Jeff Atwood. – Adam S May 25 '13 at 23:21

756 When someone asks a question on here that yes, does have a lot of links in Google about it, what they are really asking is for someone to answer the question with **plain language** and to put it in **context**, such as why do we need it, who will use it most, do we need to know it to build Android apps in Android Studio, etc. Because most often the links you find about it on Google are in "tech-speak" and for a beginner this is confusing. – Azurespot May 19

gradle just the background process that compiles the code I have written? Also, after looking at lots of other answers, they mention dependencies. What are those? THanks so much! – user5562706 Jan 24 '16 at 1:02

May look at concise and Nice explanation here Introducing gradle. Quite helpful – eRaisedToX Sep 18 '16 at 7:02

20 Answers

1411

Short Answer

Gradle is a build system.

Long Answer

Before Android Studio you were using Eclipse for your development purposes, and, chances are, you didn't know how to build your Android APK without Eclipse.

You can do this on the command line, but you have to learn what each tool (dx, aapt) does in the SDK. Eclipse saved us all from these low level but important, fundamental details by giving us their own build system.

Now, have you ever wondered why the `res` folder is in the same directory as your `src` folder?

This is where the build system enters the picture. The build system automatically takes all the source files (`.java` or `.xml`), then applies the appropriate tool (e.g. takes `java` class files and converts them to `dex` files), and groups all of them into one compressed file, our beloved APK.

This build system uses some conventions: an example of

(in Eclipse it is `\src` folder) or resources files (in Eclipse it is `\res` folder).

Now, in order to automate all these tasks, there has to be a script; you can write your own build system using shell scripting in linux or batch files syntax in windows. Got it?

Gradle is another **build system** that takes the best features from other build systems and combines them into one. It is improved based off of their shortcomings. It is a **JVM based build system**, what that means is that you can write your own script in Java, which Android Studio makes use of.

One cool thing about gradle is that it is a **plugin based system**. This means if you have your own programming language and you want to automate the task of building some package (output like a JAR for Java) from sources then you can write a complete plugin in Java or Groovy(or Kotlin, see here), and distribute it to rest of world.

Why did Google use it?

Google saw one of the most advanced build systems on the market and realized that you could write scripts of your own with little to no learning curve, and without learning Groovy or any other new language. So they wrote the Android plugin for Gradle.

You must have seen `build.gradle` file(s) in your project. That is where you can write scripts to automate your tasks. The code you saw in these files is Groovy code. If you write `System.out.println("Hello Gradle!");` then it will print on your console.

What can you do in a build script?

process happens. A Gradle build script can do this.

edited Jun 19 '17 at 11:07

Trent Oh
129 2 6

answered Jul 18 '14 at 15:27

Gagandeep Singh
14.6k 1 12 17

11 I have to say that Gradle was giving me issues when taking an AndroidStudio project from one location to another. Mismatches of versions, missing modules, whatnot... it was giving me hard time. I ended up re-creating the project skeleton and cutting/pasting the parts of code that were from the old project to the new one. And this is under Windows for both sources and destination. I am going to move it now under a Linux-based lab. – jfmessier Jan 19 '15 at 19:03

14 What I don't understand is why gradle tries to connect to Internet for every single project that I make or even every single time I want to compile – AaA Mar 4 '15 at 3:23

2 @user1700737 : To generate simple apk android studio is provided over gradle so if you ask me then there is no need to go into internals of gradle as far as simple apk generation is cencerned. Gradle is build system and helps to achieve complex,dependent tasks in one go and automated way. – Gagandeep Singh Mar 12 '15 at 5:27

4 I thought IntelliJ already knew how to build Android projects without the need for another build system. – Arne Evertsson Aug 18 '15 at 12:57

16 Another key advantage of gradle over the pre Android studio build system(s) is dependency management. Previously to use an Android library in a project, one would have to download the source code, add it to your project and compile it with your project. Instead, with gradle one can add a single line to the build.gradle and gradle will then download the compiled library from a public repository and add it to your project. Some examples:

This also works for standard Java jar files instead of putting them in the lib directory. – Stan Kurdziel Oct 28 '15 at 16:11

151

It's the new build tool that Google wants to use for Android. It's being used due to it being more extensible, and useful than ant. It is meant to enhance developer experience.

You can view a talk by Xavier Ducrohet from the Android Developer Team on Google I/O here.

There is also another talk on Android Studio by Xavier and Tor Norbye, also during Google I/O here.

answered May 26 '13 at 3:48

daniel_c05
7,520 12 54 76

3 I thought IntelliJ already knew how to build Android projects without the need for another build system. – Arne Evertsson Aug 18 '15 at 13:02

1 So is gradle just the background process that compiles the code I have written? Also, after looking at lots of other answers, they mention dependencies. What are those? THanks so much daniel! – user5562706 Jan 24 '16 at 1:01

1 @user5562706 It doesn't compile the code, but it's responsible for calling the executable that does compile your code. After compiling your code, it can then run another task to package it up, another to copy it to a directory of your choosing, then maybe another to upload it to a server somewhere... It's those sorts of tasks Gradle, or any build system, will do. – Michael Berry Jun 26 '18 at 10:43

Gradle is a build system running on **Android Studio**.

- Ant and Maven of **Java**
- Rake of **Ruby**
- A-A-P of **C**
- NAnt of **.NET**
- Make in **Linux**

answered Aug 21 '15 at 2:25

itiskj
3,863 1 28 38

13 When you said `Gradle` is similar to `make` , I just got its concept – user3405291 Dec 13 '17 at 11:33

Simple answer !!! – Kalanidhi Jul 12 '18 at 14:18

2 PHING of **PHP** – Yousha Aleayoub Jul 25 '18 at 17:34

9 One of the fastest ways I've found to learn what an unknown tool does and where it falls in the pipeline, is to type into Google the name of the tool and "vs". The results will usually show "X vs Y" where Y is some other tool you might be more familiar with. – James Kuang Sep 7 '18 at 15:51

38

Here is a detailed explanation about what `Gradle` is and how to use it in Android Studio.

Exploring the Gradle Files

1. Whenever you create a project in Android Studio, the build system automatically generates all the necessary Gradle build files.

Gradle Build Files

2. Gradle build files use a `Domain Specific Language or`

3. Android Studio projects consists of 1 or more modules, which are components that you can build, test, and debug independently. Each module has its own build file, so every Android Studio project contains 2 kinds of Gradle build files.
4. **Top-Level Build File:** This is where you'll find the configuration options that are common to all the modules that make up your project.
5. **Module-Level Build File:** Each module has its own Gradle build file that contains module-specific build settings. You'll spend most of your time editing module-level build file(s) rather than your project's top-level build file.

To take a look at these `build.gradle` files, open Android Studio's Project panel (by selecting the Project tab) and expand the Gradle Scripts folder. The first two items in the Gradle Scripts folder are the project-level and module-level Gradle build files

Top-Level Gradle Build File

Every Android Studio project contains a single, top-level Gradle build file. This `build.gradle` file is the first item that appears in the Gradle Scripts folder and is clearly marked Project.

> Most of the time, you won't need to make any changes to this file, but it's still useful to understand its contents and the role it plays within your project.

Module-Level Gradle Build Files

In addition to the project-level Gradle build file, each module has a Gradle build file of its own. Below is an annotated version of a basic, module-level Gradle build

Other Gradle Files

In addition to the build.gradle files, your Gradle Scripts folder contains some other Gradle files. Most of the time you won't have to manually edit these files as they'll update automatically when you make any relevant changes to your project. However, it's a good idea to understand the role these files play within your project.

gradle-wrapper.properties (Gradle Version)

This file allows other people to build your code, even if they don't have Gradle installed on their machine. This file checks whether the correct version of Gradle is installed and downloads the necessary version if necessary.

settings.gradle

This file references all the modules that make up your project.

gradle.properties (Project Properties)

> This file contains configuration information for your entire project. It's empty by default, but you can apply a wide range of properties to your project by adding them to this file.

local.properties (SDK Location)

This file tells the Android Gradle plugin where it can find your Android SDK installation.

Note: `local.properties` contains information that's specific to the local installation of the Android SDK. This means that you shouldn't keep this file under source control.

Suggested reading - Tutsplus Tutorial

edited Oct 11 '18 at 1:33

Mirmuhsin Sodiqov
61 2 11

answered Dec 12 '15 at 8:04

Chavan Parmeshwar
599 1 8 20

1 Note that link-only answers are discouraged, SO answers should be the end-point of a search for a solution (vs. yet another stopover of references, which tend to get stale over time). Please consider adding a stand-alone synopsis here, keeping the link as a reference. – kleopatra Dec 12 '15 at 9:00

29

Gradle is one type of **build tool** that builds the source code of the program. So it's an important part of Android Studio, and needs to be installed before starting developing your application.

We do not have to install it separately, because the Android Studio does it for us, when we make our first project.

edited Nov 16 '15 at 4:38

hichris123
7,540 12 43 63

answered Jan 19 '14 at 9:14

karan
291 3 5

2 What do you mean by building the source code of the program? Isn't that the job of a developer? – committedandroider Dec 2 '14 at 8:31

code is always being built by the compiler and nothing else. – AutomEng May 31 '15 at 18:15

Yet from another question it sounds like Gradle won't work properly without a reliable internet conneciton, does this mean I have to throw away Android Studio until that is fixed? – Michael Jun 30 '15 at 0:27

1 @committedandroider: the job of a developer is to *write* the source code, not *build* it. In system developement we use the word "Build" to mean the process of *packaging* the *compiled* source code into a final *product* (a *package* or an *executable*) – MestreLion Mar 2 '18 at 8:04

1 @MestreLion lol I was pretty confused back then because people used 'building' in reference to writing source code for their projects- i built this chat server, this app to... – committedandroider Mar 3 '18 at 0:40

23

You can find everything you need to know about Gradle here: Gradle Plugin User Guide

> ### Goals of the new Build System
>
> The goals of the new build system are:
>
> - Make it easy to reuse code and resources
> - Make it easy to create several variants of an application, either for multi-apk distribution or for different flavors of an application
> - Make it easy to configure, extend and customize the build process
> - Good IDE integration
>
> ### Why Gradle?

Here are some of its features that made us choose Gradle:

- Domain Specific Language (DSL) to describe and manipulate the build logic
- Build files are Groovy based and allow mixing of declarative elements through the DSL and using code to manipulate the DSL elements to provide custom logic.
- Built-in dependency management through Maven and/or Ivy.
- Very flexible. Allows using best practices but doesn't force its own way of doing things.
- Plugins can expose their own DSL and their own API for build files to use.
- Good Tooling API allowing IDE integration

edited Nov 16 '15 at 4:41

hichris123
7,540 12 43 63

answered Jan 21 '15 at 4:50

Mahdi Rashidi
871 14 30

23

DEFINITION:: *Gradle can be described a structured building mechanism where it provides a developer the tools and flexibility to manage the resources of a project to create builds that are* `smaller in size , targeting specific requirements` *for certain devices of certain configurations*

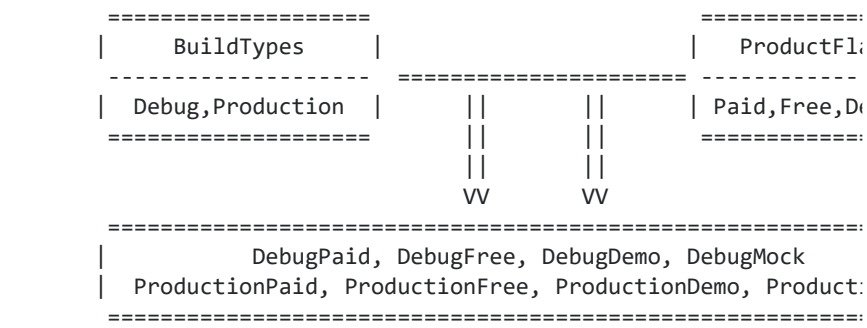
BASIC CONFIGURATIONS

2. `maximumSdk`
3. `targettedSdk`
4. `versionCode`
5. `versionName`

LIBRARIES:: *We can add android libraries or any other third party libraries in addition as per requirements easy which was a tedious task earlier. If the library does not fit for the existing project, The developer is shown a log where the person can find a appropriate solution to make changes to the project so that the library can be added. Its just one line of dependency*

GENERATING VARIETIES OF BUILDS

Combining build types with build flavours to get varities of build varients



REDUCING SIZE

Gradle helps in reducing the size of the generated build by removing the unused resources also unused things from

MANAGING PERMISSIONS

We can Specify certain permissions for certain builds by adding certain permissions in certain scenarios based on requirements

BUILDS FOR CERTAIN DEVICES

We can manage generating build for certain devices that include certain densities and certain api levels. This helps in product deployments in app store according to requirements across multiple types of devices

GOOD REFERENCE

Vogella Tutorials

edited Nov 26 '16 at 17:43

answered Nov 26 '16 at 17:21

Devrath

23.1k 41 133 191

21

Gradle is a build system. **Build systems** are software tools designed to automate the process of program compilation. Build systems come in various forms, and are used for a variety of software build tasks. While their primary goal is to efficiently create executables.

Another related term is Build automation which is the

source code into binary code, packaging binary code, and running automated tests.

Few similar build system for other languages are (see complete list here):

1. Apache Ant & Apache Maven - Java
2. sbt (Simple Build Tool) - for Scala (Play framework etc)
3. A-A-P - Python based build tool
4. Rake (Apache Builder) - Ruby
5. Leiningen for Clojure

edited Aug 9 '16 at 11:58

answered Sep 18 '15 at 8:28
user4774371

16

At the risk of being discursive I think behind this is the question of why the Android Studio / Gradle experience is so bad.

Typical Clojure experience :

- download project with dependencies listed in project.clj.
- Leiningen gets the dependencies thanks to Clojars and Maven.
- Project compiles.

Typical Android Studio / Gradle experience :

- OK project imported.
- Gradle is doing it's thang ... wait ... wait ... wait ... Gradle has finished.
- Compile ... can't compile because I don't know what an X is / can't find Y library.

I'm not sure this is Gradle's fault exactly. But the "import from Eclipse project" seems pretty flaky. For all of Gradle's alleged sophistication and the virtues of a build-system, Android Studio just doesn't seem to import the build dependencies or build-process from Eclipse very well.

It doesn't tell you when it's failed to import a complete dependency graph. The Android Studio gives no useful help or tips as to how to solve the problem. It doesn't tell you where you can manually look in the Eclipse folders. It doesn't tell you which library seems to be missing. Or help you search Maven etc. for them.

In 2016 things like Leiningen / Clojars, or node's npm, or Python's pip, or the Debian apkg (and I'm sure many similar package managers for other languages and systems) all work beautifully ... missing dependencies are thing of the past.

Except with Android. Android Studio is now the only place where I still seem to experience missing-dependency hell.

I'm inclined to say this is Google's fault. They broke the Android ecosystem (and thousands of existing Android projects / online tutorials) when they cavalierly decided to shift from Eclipse to Android Studio / Gradle without producing a robust conversion process. People whose projects work in Eclipse aren't adapting them to AS (presumably because it's a pain for them). And people trying to use those projects in AS are hitting the same issues.

And anyway, if Gradle is this super-powerful build system, why am I still managing a whole lot of other dependencies in the sdk manager? Why can't a project that needs, say, the ndk specify this in its Gradle file so that it gets automatically installed and built-against when needed? Why is NDK special? Similarly for target platforms? Why am I installing them explicitly in the IDE rather than just checking my project against them and having this all sorted for me behind the scenes?

answered Aug 21 '16 at 14:33

interstar
9,098 29 86 146

4 I fully agree that Gradle is a terrible build system. It is the component that makes Android Studio far worse than X-Code or Visual Studio. I realise these must "build" the system behind the scenes but I'm not really required to know/care/tinker with their internals. Gradle on the other hand seems to break with every Android Studio update and bogs down the system even just adding a class. No other IDE does that. – csmith Oct 31 '16 at 16:01

15

I refer two tutorial to write the Answer one,two

Gradle is a general purpose, declarative build tool. It is general purpose because it can be used to build pretty much anything you care to implement in the build script. It is declarative since you don't want to see lots of code in the build file, which is not readable and less maintainable. So, while Gradle provides the idea of conventions and a simple and declarative build, it also makes the tool adaptable and developers the ability to extend. It also provides an easy way to customize the default behavior and different hooks to add any third-party features.

> Gradle combines the good parts of both tools and provides additional features and uses Groovy as a Domain Specific Language (DSL). It has power and flexibility of Ant tool with Maven features such as build life cycle and ease of use.

Why Gradle? Why Now?

The build tool's response is to add scripting functionality through nonstandard extension mechanisms. You end up mixing scripting code with XML or invoking external scripts from your build logic. It's easy to imagine that you'll need to add more and more custom code over time. As a result, you inevitably introduce accidental complexity, and maintainability goes out the window.

> Let's say you want to copy a file to a specific location when you're building the release version of your project. To identify the version, you check a string in the metadata describing your project. If it matches a specific numbering scheme (for example, 1.0-RELEASE), you copy the file from point A to point B. From an outside perspective, this may sound like a trivial task. If you have to rely on XML, the build language of many traditional tools, expressing this simple logic becomes fairly difficult.

Evolution of Java Build Tools

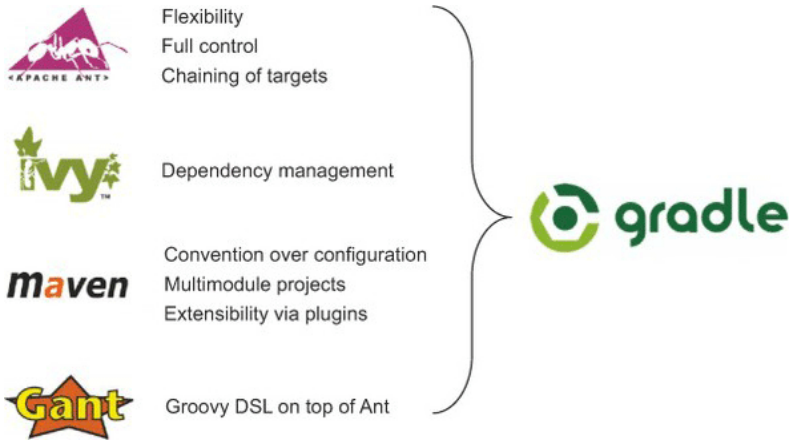
> Java build logic has to be described in XML. XML is great for describing hierarchical data but falls short on expressing program flow and conditional logic. As a build script grows in complexity, maintaining the building code becomes a nightmare.

your project. Though it allows for maximum flexibility, Ant makes each build script unique and hard to understand. External libraries required by your project are usually checked into version control because there is no automated mechanism to pull them from a central location.

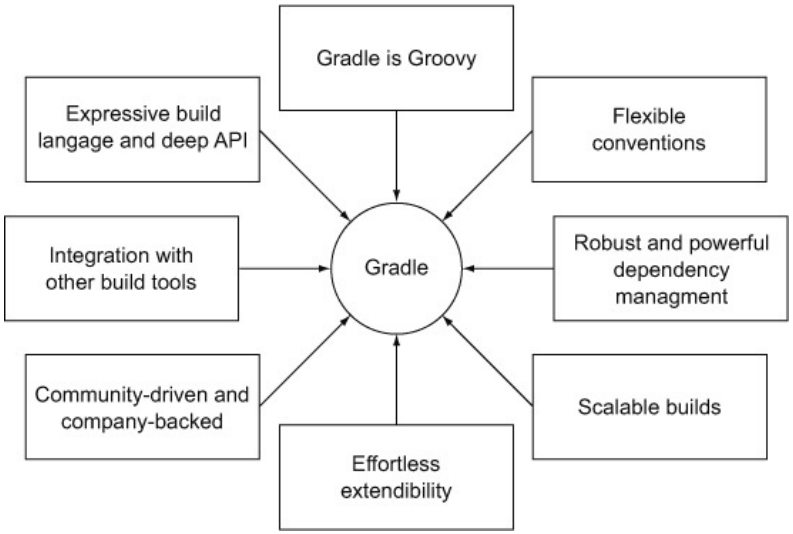
Maven 1, released in July 2004, tried to ease that process. It provided a standardized project and directory structure, as well as dependency management. Unfortunately, custom logic is hard to implement

Gradle fits right into that generation of build tools and satisfies many requirements of modern build tools (Figure 1). It provides an expressive DSL, a convention over configuration approach, and powerful dependency management. It makes the right move to abandon XML and introduce the dynamic language Groovy to define your build logic. Sounds compelling, doesn't it?

Gradle combines the best features of other build tools.



Gradle's Compelling Feature Set

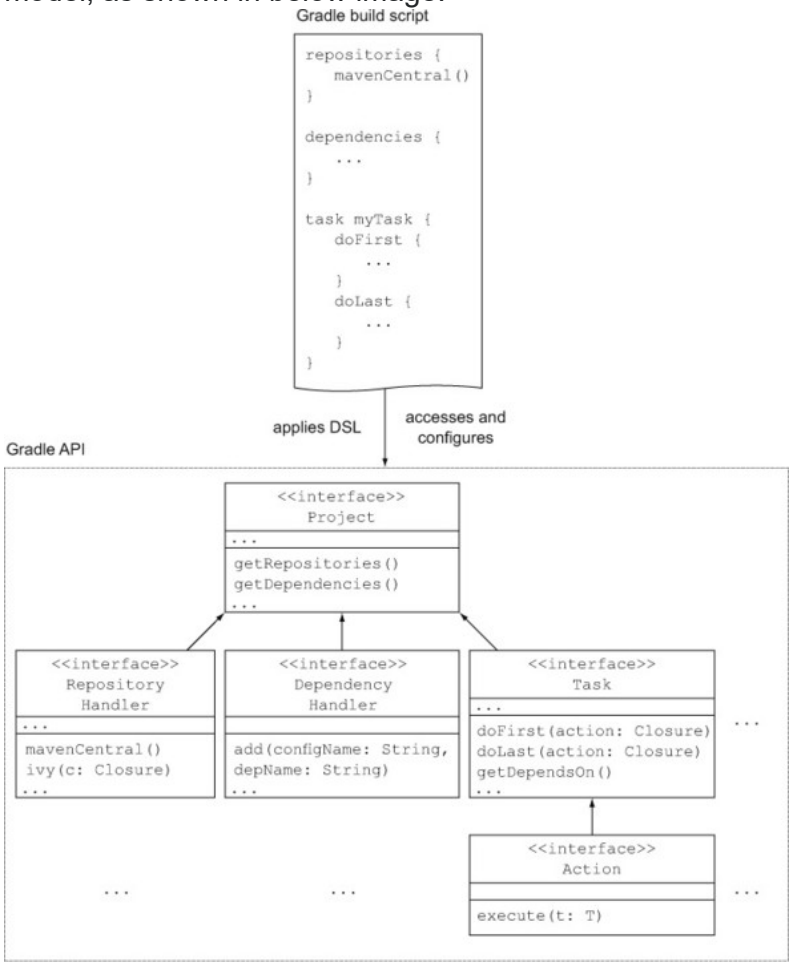


Why Build Your Java Projects with Gradle Rather than Ant or Maven?

> **The default build tool for Android (and the new star of build tools on the JVM) is designed to ease scripting of complex, multi-language builds. Should you change to it, though, if you're using Ant or Maven?**

The key to unlocking Gradle's power features within your build script lies in discovering and applying its domain

model, as shown in below image.



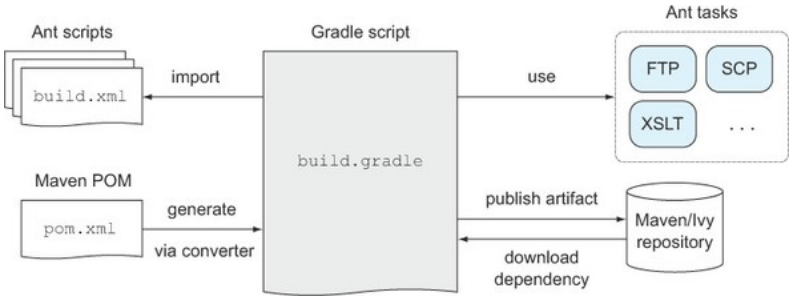
Gradle can't know all the requirements specific to your enterprise build. By exposing hooks into lifecycle phases, Gradle allows for monitoring and configuring the build script's execution behavior.

> **Gradle establishes a vocabulary for its model by exposing a DSL implemented in Groovy. When dealing with a complex problem domain, in this case, the task of building software, being able to**

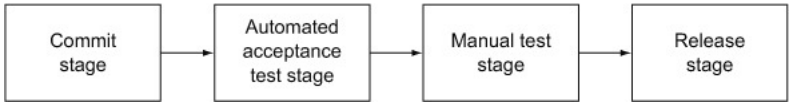
> Another example is the way you can express dependencies to external libraries, a very common problem solved by build tools. Out-of-the-box Gradle provides you with two configuration blocks for your build script that allow you to define the dependencies and repositories that you want to retrieve them from. If the standard DSL elements don't fit your needs, you can even introduce your own vocabulary through Gradle's extension mechanism.

Integration with Other Build Tools

Gradle plays well with its predecessors' Ant, Maven, and Ivy, as shown in the image below.



Automating Your Project from Build to Deployment



In image: Stages of a deployment pipeline.

- Compiling the code
- Running unit and integration tests
- Performing static code analysis and generating test coverage
- Creating the distribution

- Deploying the deliverable
- Performing smoke and automated functional tests

I refer two tutorial to write the Answer one,two

edited Feb 13 at 11:52

Abhishek Oza
2,154 1 21 31

answered Mar 14 '18 at 6:19

Farhana
2,799 6 24 35

1 In addition to that, You can build your Android apps from the command line on your machine or on machines where Android Studio is not installed (such as continuous integration servers). – jettimadhuChowdary Mar 27 '18 at 5:07

13

`Gradle` is an advanced build system as well as an advanced build toolkit allowing to create custom build logic through plugins!

Advantages:

- Dsl - Domain specific language, based on groovy
- DAG - Directed Acyclic Graph
- Incremental builds
- Extensible domain model
- Gradle is always up to date
- Before a task is being execute, Gradle takes a snapshot of its task's input and output.
- In case the snapshot has changed or it doesn't exists,

Through the DSL it is possible to configure the following manifest entries:

Build variant

By default, the Android plugin automatically sets up the project to build both a debug and a release version of the application.

Dependencies

1. Local Dependencies:

> If you have binary archives in your local filesystem that a module depends on, such as JAR files, you can declare these dependencies in the build file for that module.

2. Remote Dependencies:

> First the repository must be added to the list, and then the dependency must be declared in a way that Maven or Ivy declare their artifacts.

edited Apr 11 '18 at 5:50

Rohit Sharma
692 2 9 27

answered Nov 17 '16 at 10:03

SKallmann
145 1 5

Gradle is to the Groovy JVM language what ant is to Java. Basically, it's Groovy's build tool. Unlike Ant, it's based on the full Groovy language. You can, for example, write

I don't know IntelliJ's specific integration, but imagine you could "extend" Groovy such that you could write specific "build" language primitives and they just became part of the Groovy language. (Groovy's metaprogramming is a whole discussion unto itself.) IntelliJ/Google could use Gradle to build a very high-level build language, yet, it's a language build on an expandable, open standard.

answered May 26 '13 at 1:34

user500123
535 4 10

2 it's not "Groovys build tool" - it's a multi purpose build tool whose DSL is build on top of groovy – martyglaubitz Dec 19 '13 at 18:41

9

Gradle is an automated build toolkit that can integrate into lots of different environments not only for Android projects.

Here are few things that you can do with gradle.

- **Minimal Configuration Required for New Projects because Gradle has defaults configurations for your android studio projects.**
- **Dependancy Declaration. You can declare dependency jar files or library files that is hosted in local or remote server.**
- **Gradle automatically generates a test directory and a test APK from your project's source.**
- **If you add all the necessary information, such as `keyPassword` and `keyAlias`, to your Gradle build file, you can use Gradle to generate signed APKs.**
- **Gradle can generate multiple APKs with different**

edited Mar 20 '17 at 19:57

Jad Chahine
3,133 4 23 45

answered Apr 28 '16 at 6:11

Zumry Mohamed
5,214 2 25 39

8

In Android Studio, Gradle is a custom build tool used to build android packages (apk files) by managing dependencies and providing custom build logic.

APK file (Android Application package) is a specially formatted zip file which contains

- Byte code
- Resources (images, UI, xml etc)
- Manifest file

An apk file gets signed and pushed to the device using ADB(Android Debug Bridge) where it gets executed.

edited Sep 18 '15 at 8:17

Valery Viktorovsky
4,926 3 25 37

answered Apr 18 '15 at 7:13

Akash Chandwani
382 1 7 16

Gradle is an advanced build toolkit for android that manages dependencies and allows you to define custom build logic, features are like

- Create multiple APKs for your app with different features using the same project.
- Reuse code and resources.

refer

edited Nov 23 '18 at 9:31

Agilanbu
1,282 14 20

answered May 23 '14 at 10:50

AndEngine
456 1 11 26

6

Gradle is a build tool custom and used for building APK or known as application package kit.

answered Jul 24 '16 at 5:11

Samarth Shah
109 2 2

4

by **@Brian Gardner:**

Gradle is an extensive build tool and dependency manager for programming projects. It has a domain specific language based on Groovy. Gradle also provides build-by-convention support for many types of projects including Java, Android and Scala.

Feature of Gradle:

1. Dependency Management
2. Using Ant from Gradle

4. Java Plugin
5. Android Plugin
6. Multi-Project Builds

answered Apr 23 '16 at 14:37

Dhaval Jivani
5,924 1 30 32

3

Gradle is what makes it possible to automate the building of complex Android projects that involve 10s of thousands of lines of code from multiple sources, projects, libraries etc. It can conditionally generate multiple optimised APKs based on a plethora of configuration specifications - if you are interested, the other answers provide more details of this aspect of Gradle.

However, if you're new to Android developing, Gradle in 99% of cases is what **stops** your project from building. It is an inscrutable, complex system that effectively obfuscates the Android build process and essentially renders it unavailable to inexperienced developers, ie in order to build a simple entry level Android App the unsuspecting newbie might need to study and understand many things that they didnt bargain for such as:

- Android APK structure and ecosystem
- Android Studio
- Java Classpaths and dependencies
- Groovy
- Gradle build scripts
- Many other complex and interesting technologies

the OP to ask this question is the feeling of frustration that inevitably hits the neophyte developer after spending way too long trying to get a simple app to build and being continually thwarted by Gradle. The problem is perversely exacerbated by the overwhelming quantity of highly technical documentation that is available for all these technologies. Also for a large amount of development needs Gradle is overkill.

An alternative is to write a shell script that builds your project by automating the tools available in the android SDK. The virtues of this approach are many, for starters its probably the best way to study and understand the build process and the Android ecosystem, and it allows you to completely control how your app is built. However this approach is more suitable for deeply irredeemable tech-heads than it is to inexperienced noobs trying out android.

What is conspicuous by its absence (please inform me if there is such a thing) is an entry level, lightweight IDE with a reduced feature set that simultaneously simplifies the build process while not obscuring it (so not netbeans or eclipse) it could possibly still use Gradle (what was wrong with Ant). It should make it easy to generate APKs that conform to a few common configurations and use a project structure that can evolve to a full Android Studio project should you decide to take it that way.

answered Nov 26 '17 at 3:03

flurbius
810 6 14

1

In plain terms, Gradle is a tool provided by Android Studio in order to implement two important processes:

2. Package AndroidManifest.xml,res folder,and binary code into a specially formatted zip file called APK

answered Dec 8 '17 at 22:55

S.Voulgaris
21 1

0

Gradle = Groovy + Cradle [Hans Dockter forum comment]

The confusion is a bit unnecessary when it could have just been called "Build" or something in Android Studio.

We like to make things difficult for ourselves in the Development community.

answered Mar 8 '18 at 14:57

James_UK_DEV
214 4 14

---

protected by Community ♦ Dec 2 '14 at 12:36

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?