Gradle

* Ant uses lot of xml. So Maven came. All maven projects are similar structure . it reduces dependency issues with pom.xml. But when customization required , maven is little harder. That is where gradle came into picture .
* Gradle uses programming language like groovy or kotlin for build script.
* Gradle is a standard build tool for android .
* Gradle can work with maven repo. It will cache them locally and use it in future builds.
* In maven we can specify only the version of dependency . In gradle also specify custom rules for dependency.
* Gradle supports incremental builds .
* If we are running multiple builds parallely , it can pick objects built in on run for other builds.
* In windows put gradlew.bat your command . in linux it’s only gradlew.
* Settings . gradle allows to change project name and modify other settings.
* Gradle wrapper will pull project specific gradle version for a project. Each project can have it’s own version.

c:\vault\GitHub\Gradle\_test\test\_project>gradle.bat

Starting a Gradle Daemon (subsequent builds will be faster)

> Task :help

Welcome to Gradle 7.4.1.

Directory 'C:\vault\GitHub\Gradle\_test\test\_project' does not contain a Gradle build.

To create a new build in this directory, run gradle init

For more detail on the 'init' task, see https://docs.gradle.org/7.4.1/userguide/build\_init\_plugin.html

For more detail on creating a Gradle build, see https://docs.gradle.org/7.4.1/userguide/tutorial\_using\_tasks.html

To see a list of command-line options, run gradle --help

For more detail on using Gradle, see https://docs.gradle.org/7.4.1/userguide/command\_line\_interface.html

For troubleshooting, visit https://help.gradle.org

BUILD SUCCESSFUL in 12s

1 actionable task: 1 executed

c:\vault\GitHub\Gradle\_test\test\_project>dir

Volume in drive C is OS

Volume Serial Number is 06EB-F968

Directory of c:\vault\GitHub\Gradle\_test\test\_project

14-03-2022 17:41 <DIR> .

14-03-2022 17:41 <DIR> ..

0 File(s) 0 bytes

2 Dir(s) 203,632,459,776 bytes free

c:\vault\GitHub\Gradle\_test\test\_project>gradle init

Select type of project to generate:

1: basic

2: application

3: library

4: Gradle plugin

Enter selection (default: basic) [1..4] 1

Select build script DSL:

1: Groovy

2: Kotlin

Enter selection (default: Groovy) [1..2] 1

Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no] no

Project name (default: test\_project): test\_project

> Task :init

Get more help with your project: Learn more about Gradle by exploring our samples at https://docs.gradle.org/7.4.1/samples

BUILD SUCCESSFUL in 1m 12s

2 actionable tasks: 2 executed

<-------------> 0% WAITING

> IDLE

c:\vault\GitHub\Gradle\_test\test\_project>code

'code' is not recognized as an internal or external command,

operable program or batch file.

c:\vault\GitHub\Gradle\_test\test\_project>gradle task

> Task :tasks

------------------------------------------------------------

Tasks runnable from root project 'test\_project'

------------------------------------------------------------

Build Setup tasks

-----------------

init - Initializes a new Gradle build.

wrapper - Generates Gradle wrapper files.

Help tasks

----------

buildEnvironment - Displays all buildscript dependencies declared in root project 'test\_project'.

dependencies - Displays all dependencies declared in root project 'test\_project'.

dependencyInsight - Displays the insight into a specific dependency in root project 'test\_project'.

help - Displays a help message.

javaToolchains - Displays the detected java toolchains.

outgoingVariants - Displays the outgoing variants of root project 'test\_project'.

projects - Displays the sub-projects of root project 'test\_project'.

properties - Displays the properties of root project 'test\_project'.

tasks - Displays the tasks runnable from root project 'test\_project'.

To see all tasks and more detail, run gradle tasks --all

To see more detail about a task, run gradle help --task <task>

BUILD SUCCESSFUL in 3s

1 actionable task: 1 executed

c:\vault\GitHub\Gradle\_test\test\_project>gradle task

> Configure project :

Hello World

> Task :tasks

------------------------------------------------------------

Tasks runnable from root project 'test\_project'

------------------------------------------------------------

Build Setup tasks

-----------------

init - Initializes a new Gradle build.

wrapper - Generates Gradle wrapper files.

Help tasks

----------

buildEnvironment - Displays all buildscript dependencies declared in root project 'test\_project'.

dependencies - Displays all dependencies declared in root project 'test\_project'.

dependencyInsight - Displays the insight into a specific dependency in root project 'test\_project'.

help - Displays a help message.

javaToolchains - Displays the detected java toolchains.

outgoingVariants - Displays the outgoing variants of root project 'test\_project'.

projects - Displays the sub-projects of root project 'test\_project'.

properties - Displays the properties of root project 'test\_project'.

tasks - Displays the tasks runnable from root project 'test\_project'.

To see all tasks and more detail, run gradle tasks --all

To see more detail about a task, run gradle help --task <task>

BUILD SUCCESSFUL in 7s

1 actionable task: 1 executed

c:\vault\GitHub\Gradle\_test\test\_project>gradle firstTask

> Configure project :

Hello World

BUILD SUCCESSFUL in 2s

c:\vault\GitHub\Gradle\_test\test\_project>

* Creating java project and build

c:\vault\GitHub\Gradle\_test\test\_java\_project>gradle init

Select type of project to generate:

1: basic

2: application

3: library

4: Gradle plugin

Enter selection (default: basic) [1..4] 2

Select implementation language:

1: C++

2: Groovy

3: Java

4: Kotlin

5: Scala

6: Swift

Enter selection (default: Java) [1..6] 3

Split functionality across multiple subprojects?:

1: no - only one application project

2: yes - application and library projects

Enter selection (default: no - only one application project) [1..2] 1

Select build script DSL:

1: Groovy

2: Kotlin

Enter selection (default: Groovy) [1..2] 1

Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no]

Select test framework:

1: JUnit 4

2: TestNG

3: Spock

4: JUnit Jupiter

Enter selection (default: JUnit Jupiter) [1..4] 1

Project name (default: test\_java\_project):

Source package (default: test\_java\_project): io.gtihub.rwik

> Task :init

Get more help with your project: https://docs.gradle.org/7.4.1/samples/sample\_building\_java\_applications.html

BUILD SUCCESSFUL in 4h 24m 3s

2 actionable tasks: 2 executed

c:\vault\GitHub\Gradle\_test\test\_java\_project>gradle tasks --all

> Task :tasks

------------------------------------------------------------

Tasks runnable from root project 'test\_java\_project'

------------------------------------------------------------

Application tasks

-----------------

app:run - Runs this project as a JVM application

Build tasks

-----------

app:assemble - Assembles the outputs of this project.

app:build - Assembles and tests this project.

app:buildDependents - Assembles and tests this project and all projects that depend on it.

app:buildNeeded - Assembles and tests this project and all projects it depends on.

app:classes - Assembles main classes.

app:clean - Deletes the build directory.

app:jar - Assembles a jar archive containing the main classes.

app:testClasses - Assembles test classes.

Build Setup tasks

-----------------

init - Initializes a new Gradle build.

wrapper - Generates Gradle wrapper files.

Distribution tasks

------------------

app:assembleDist - Assembles the main distributions

app:distTar - Bundles the project as a distribution.

app:distZip - Bundles the project as a distribution.

app:installDist - Installs the project as a distribution as-is.

Documentation tasks

-------------------

app:javadoc - Generates Javadoc API documentation for the main source code.

Help tasks

----------

buildEnvironment - Displays all buildscript dependencies declared in root project 'test\_java\_project'.

app:buildEnvironment - Displays all buildscript dependencies declared in project ':app'.

dependencies - Displays all dependencies declared in root project 'test\_java\_project'.

app:dependencies - Displays all dependencies declared in project ':app'.

dependencyInsight - Displays the insight into a specific dependency in root project 'test\_java\_project'.

app:dependencyInsight - Displays the insight into a specific dependency in project ':app'.

help - Displays a help message.

app:help - Displays a help message.

javaToolchains - Displays the detected java toolchains.

app:javaToolchains - Displays the detected java toolchains.

outgoingVariants - Displays the outgoing variants of root project 'test\_java\_project'.

app:outgoingVariants - Displays the outgoing variants of project ':app'.

projects - Displays the sub-projects of root project 'test\_java\_project'.

app:projects - Displays the sub-projects of project ':app'.

properties - Displays the properties of root project 'test\_java\_project'.

app:properties - Displays the properties of project ':app'.

tasks - Displays the tasks runnable from root project 'test\_java\_project' (some of the displayed tasks may belong to subprojects).

app:tasks - Displays the tasks runnable from project ':app'.

Verification tasks

------------------

app:check - Runs all checks.

app:test - Runs the test suite.

Other tasks

-----------

app:compileJava - Compiles main Java source.

app:compileTestJava - Compiles test Java source.

components - Displays the components produced by root project 'test\_java\_project'. [deprecated]

app:components - Displays the components produced by project ':app'. [deprecated]

dependentComponents - Displays the dependent components of components in root project 'test\_java\_project'. [deprecated]

app:dependentComponents - Displays the dependent components of components in project ':app'. [deprecated]

model - Displays the configuration model of root project 'test\_java\_project'. [deprecated]

app:model - Displays the configuration model of project ':app'. [deprecated]

prepareKotlinBuildScriptModel

app:processResources - Processes main resources.

app:processTestResources - Processes test resources.

app:startScripts - Creates OS specific scripts to run the project as a JVM application.

BUILD SUCCESSFUL in 4s

1 actionable task: 1 executed

c:\vault\GitHub\Gradle\_test\test\_java\_project>gradle compileJava

BUILD SUCCESSFUL in 7s

1 actionable task: 1 executed

c:\vault\GitHub\Gradle\_test\test\_java\_project>gradle test

BUILD SUCCESSFUL in 5s

3 actionable tasks: 2 executed, 1 up-to-date

c:\vault\GitHub\Gradle\_test\test\_java\_project>

* In build.gradle we define dependencies only for building and running ut. These comes from repository . from where libraries are pulled.
* You can mark a dependency for compile time or runtime. If you mark compile time, then those wont be available during runtime.
* Mark as api or implementation inside dependencies to control scope of libs.
* Gardel phases = initialization (it will read build.gradle and create a object of that) , configuration ( creates an acyclic graph of tasks), execution (compiled / build and jar/war created).
* doFirst () = will be done at beginning of execution
* doLast() = will be done at end of execution
* If any command given outside task , then it will be executed during configuration
* Gradle build --daemon . will make future builds fast.
* If you dont want to use daemon then gradle build --no-daemon
* A plugin is nothing but a set of tasks. It will be .groovy file.