|  |  |  |
| --- | --- | --- |
| ANT | MAVEN | GRADDLE |
| Ant **doesn't has formal conventions**, so we need to provide information of the project structure in build.xml file. | Maven **has a convention** to place source code, compiled code etc. So we don't need to provide information about the project structure in pom.xml file.  Maven’s performance is slower than Gradle’s | Gradle’s construction time is short and fast |
| Ant is **procedural**, you need to provide information about what to do and when to do through code. You need to provide order. | Maven is **declarative**, everything you define in the pom.xml file.  Maven’s scripts are slightly longer than Gradle’s | Gradle’s scripts are much shorter and cleaner |
| There is **no life cycle** in Ant. | There is **life cycle** in Maven.  It uses XML | It uses domain specific language (DSL) |
| It is **a tool** box. | It is **a framework**.  In Maven, objectives linked to the project are defined | It is based on the task through which the work carried out |
| It is **mainly a build tool**. | It is **mainly a project management tool**.  It does not support incremental compilations | It supports Java class incremental compilations |
| The ant scripts are **not reusable**. | The maven plugins are **reusable**.  Support for most continuous integration tools | Support for continuous integration tools |
| It is **less preferred** than Maven. | It is **more preferred** than Ant. |  |
|  |  |  |

| Basis | Gradle | Maven |
| --- | --- | --- |
| Based on | Gradle is based on developing domain-specific language projects. | Maven is based on developing pure Java language-based software. |
| Configuration | It uses a Groovy-based Domain-specific language(DSL) for creating project structure. | It uses Extensible Markup Language(XML) for creating project structure. |
| Focuses on | Developing applications by adding new features to them. | Developing applications in a given time limit. |
| Performance | It performs better than maven as it optimized for tracking only current running task. | It does not create local temporary files during software creation, and is hence – slower. |
| Java Compilation | It avoids compilation. | It is necessary to compile. |
| Usability | It is a new tool, which requires users to spend a lot of time to get used to it. | This tool is a known tool for many users and is easily available. |
| Customization | This tool is highly customizable as it supports a variety of IDE’s. | This tool serves a limited amount of developers and is not that customizable. |
| Languages supported | It supports software development in Java, C, C++, and Groovy. | It supports software development in Java, Scala, C#, and Ruby. |

| Basis | Gradle | Maven |
| --- | --- | --- |
| Based on | Gradle is based on developing domain-specific language projects. | Maven is based on developing pure Java language-based software. |
| Configuration | It uses a Groovy-based Domain-specific language(DSL) for creating project structure. | It uses Extensible Markup Language(XML) for creating project structure. |
| Focuses on | Developing applications by adding new features to them. | Developing applications in a given time limit. |
| Performance | It performs better than maven as it optimized for tracking only current running task. | It does not create local temporary files during software creation, and is hence – slower. |
| Java Compilation | It avoids compilation. | It is necessary to compile. |
| Usability | It is a new tool, which requires users to spend a lot of time to get used to it. | This tool is a known tool for many users and is easily available. |
| Customization | This tool is highly customizable as it supports a variety of IDE’s. | This tool serves a limited amount of developers and is not that customizable. |
| Languages supported | It supports software development in Java, C, C++, and Groovy. | It supports software development in Java, Scala, C#, and Ruby. |