**Maven:**

Maven is a, it is for defining how .java files get exported to .jar files. It is highly independent development tool which downloads necessary additional tools or scripts by incorporating other common tasks like downloading & installing necessary libraries etc. It is built portability theme so that the problems of code executing in one machine and not in the other is solved by using this development tool. Because of this, it is also the best way to work on a project between people who use different IDEs since IDE-generated Ant scripts are hard to import into other IDEs, but all IDEs nowadays understand and support Maven ([IntelliJ](http://www.jetbrains.com/idea/features/ant_maven.html), [Eclipse](http://www.eclipse.org/m2e/), and [NetBeans](http://wiki.netbeans.org/Maven)).

1. Maven will download all the libraries that you use and the libraries that they use for you automatically
2. You don't need to write a "compile", "test", "package", or "clean" step like you would have to do in Ant or a Makefile.

**Gradle:**

Gradle combines good parts of both tools and builds on top of them with DSL and other improvements. It has Ant’s power and flexibility with Maven’s life-cycle and ease of use. The end result is a tool that was released in 2012 and gained a lot of attention in a short period of time. For example, Google adopted Gradle as the default build tool for the Android OS.

Gradle does not use XML. Instead, it had its own DSL based on Groovy (one of JVM languages). As a result, Gradle build scripts tend to be much shorter and clearer than those written for Ant or Maven. The amount of boilerplate code is much smaller with Gradle since its DSL is designed to solve a specific problem: move software through its life cycle, from compilation through static analysis and testing until packaging and deployment.

Initially, Gradle used Apache Ivy for its dependency management. Later own it moved to its own native dependency resolution engine.

Gradle effort can be summed as “convention is good and so is flexibility”.

**SBT:**

1. sbt is an [open source](https://en.wikipedia.org/wiki/Open_source) [build tool](https://en.wikipedia.org/wiki/Build_tool) for [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)) and [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) projects.
2. SBT can consume and produce maven artifacts, so you can migrate incrementally .
3. SBT build configurations are less verbose than the equivalent POM files, and they are written in Scala code, so you can express whatever you need to get build your project, for example triggering code   
   generation. SBT console mode keeps scala resident, which really improves compile times on subsequent runs. This is important for scala, which is quite slow as compared to javac.