**ASSIGNMENT 15.2**

**Q. Explain the working and the differences between Maven, Gradle and SBT in detail.**

**Solution:**

**Maven:**

-Maven is a [build automation](https://en.wikipedia.org/wiki/Build_automation) tool used primarily for [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) projects

-Maven addresses two aspects of building software: first, it describes how software is built, and second, it describes its dependencies.

- Contrary to preceding tools like [Apache Ant](https://en.wikipedia.org/wiki/Apache_Ant), it uses conventions for the build procedure, and only exceptions need to be written down.

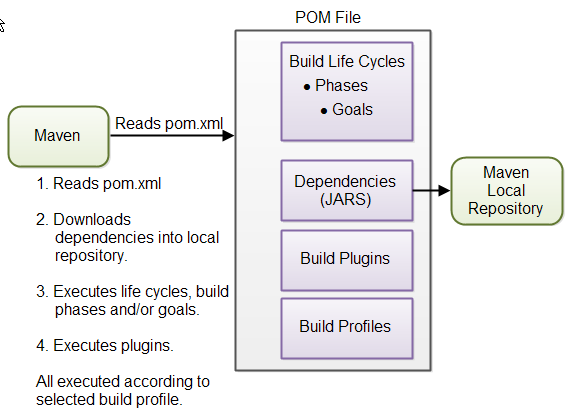
- An [XML](https://en.wikipedia.org/wiki/XML) file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required [plug-ins](https://en.wikipedia.org/wiki/Plug-in_(computing)). It comes with pre-defined targets for performing certain well-defined tasks such as compilation of code and its packaging.

-Maven dynamically downloads [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. This local cache of downloaded [artifacts](https://en.wikipedia.org/wiki/Artifact_(software_development)) can also be updated with artifacts created by local projects. Public repositories can also be updated.

-Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input. Theoretically, this would allow anyone to write plugins to interface with build tools (compilers, unit test tools, etc.) for any other language. In reality, support and use for languages other than Java has been minimal. Currently a plugin for the .NET framework exists and is maintained and a [C](https://en.wikipedia.org/wiki/C_(programming_language))/[C++](https://en.wikipedia.org/wiki/C%2B%2B) native plugin is maintained for Maven

-Maven projects are configured using a [Project Object Model](https://en.wikipedia.org/wiki/Project_Object_Model), which is stored in a pom.xml-file

-This POM only defines a unique identifier for the project (*coordinates*) and its dependency on the [JUnit](https://en.wikipedia.org/wiki/JUnit) framework. However, that is already enough for building the project and running the [unit tests](https://en.wikipedia.org/wiki/Unit_testing) associated with the project.



**Gradle:**

**-**Gradle is an advanced general purpose build management system. Gradle supports the automatic download and configuration of dependencies or other libraries. It supports Maven repositories for retrieving its dependencies. This allows reusing the artifacts of existing build systems.

- Gradle is available with separate Domain Specific Language. Gradle provides declarative language elements.

-The declarative language lies on top of a general purpose task graph, which you can fully leverage in your build.

-It allows to build the optimized structure which can be easily built.

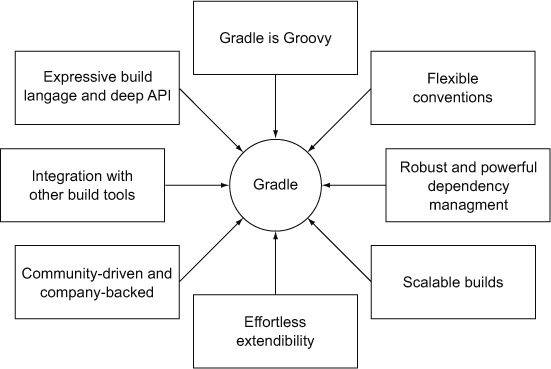
-Using Deep api and customize its configuration and execution behavior to its core.

-If you build a subproject, Gradle takes care of building all the subprojects that it depends on.

-It has different ways to manage the dependencies

-It has different ways to manage the builds.

-Gradle supports all the functions of Maven and ANT. It also provides a converter for turning a Maven pom.xml to Gradle script.



**SBT**:

-SBT (simple build tool) is an open source build tool for Scala and Java projects, similar to Java’s Maven or Ant.

-Two most prominent key features are incremental compilation and an interactive shell.

-SBT can consume and produce maven artifacts, so you can migrate incrementally .

-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.