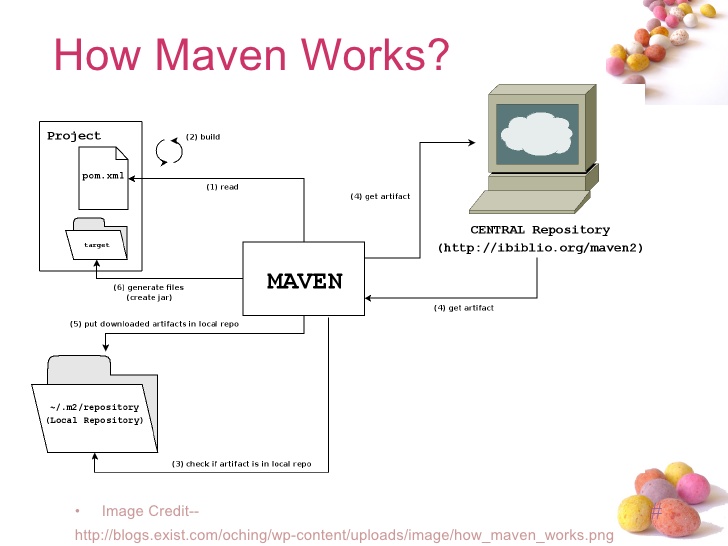
**15.2 Explain the working and the differences between Maven, Gradle and SBT in detail.**

**MAVEN:**

* Apache Maven is a software project management and comprehension tool.
* Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.
* Maven addresses two aspects of building software: first, it describes how software is built, and second, it describes its dependencies.
* Maven dynamically downloads java 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 can also be updated with artifacts created by local projects. Public repositories can also be updated.
* Maven can also be used to build and manage projects written in c#, Ruby, Scala, and other languages.
* Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input.
* This allows anyone to write plugins to interface with build tools (compilers, unit test tools, etc.) for any other language.

****

**GRADLE:**

* Gradle is an open source [build automation](https://en.wikipedia.org/wiki/Build_automation) system that builds upon the concepts of [Apache Ant](https://en.wikipedia.org/wiki/Apache_Ant) and [Apache Maven](https://en.wikipedia.org/wiki/Apache_Maven) and introduces a [Groovy](https://en.wikipedia.org/wiki/Groovy_(programming_language))-based [domain-specific language](https://en.wikipedia.org/wiki/Domain-specific_language) (DSL) instead of the [XML](https://en.wikipedia.org/wiki/XML) form used by [Apache Maven](https://en.wikipedia.org/wiki/Apache_Maven) for declaring the project configuration.
* Gradle was designed for multi-project builds which can grow to be quite large, and supports incremental builds by intelligently determining which parts of the build tree are up-to-date, so that any task dependent upon those parts will not need to be re-executed.
* With Gradle you can execute any random task in your setup and Gradle will make sure all declared dependecies are properly and timely executed. Your code can be spread across many directories in any kind of layout (tree, flat, scattered, ..,)
* Gradle has two distinct phases: evaluation and execution. Basically, during evaluation Gradle will look for and evaluate build scripts in the directories it is supposed to look.
* During execution Gradle will execute tasks which have been loaded during evaluation taking into account task inter-dependencies.
* Gradle detects dependencies between projects and between projects and JARs.
* Gradle works with Maven repositories (download and upload) like the iBiblio.
* Gradle nicely combines both Ant and Maven, taking the best from both frameworks

**ADVANCED FEATURES:**

* Parallel unit test execution
* Dependency building
* Incremental build support
* Dynamic tasks and task rules
* Gradle daemon

**SBT:**

**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, similar to Java's [Maven](https://en.wikipedia.org/wiki/Apache_Maven) or [Ant](https://en.wikipedia.org/wiki/Apache_Ant).

**MAIN FEATURES:**

* native support for compiling Scala code and integrating with many Scala [test frameworks](https://en.wikipedia.org/wiki/Test_automation)
* build descriptions written in Scala using a [DSL](https://en.wikipedia.org/wiki/Domain_Specific_Language)
* dependency management using [Ivy](https://en.wikipedia.org/wiki/Apache_Ivy) (which supports Maven-format repositories)
* continuous compilation, testing, and deployment
* integration with the Scala interpreter for rapid iteration and debugging
* support for mixed Java/Scala projects

The functionality of sbt can be extended through a plugin architecture.A dedicated website was set up for community contributed plugins, which cover various areas such as signing, packaging, publishing and releasing artifacts, connecting to other services such as blogs and databases, or integrating with other technologies such as deploying to the Android platform.There are plugins to automatically create project files for the [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software)) and [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA" \o "IntelliJ IDEA) IDEs.

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 java.