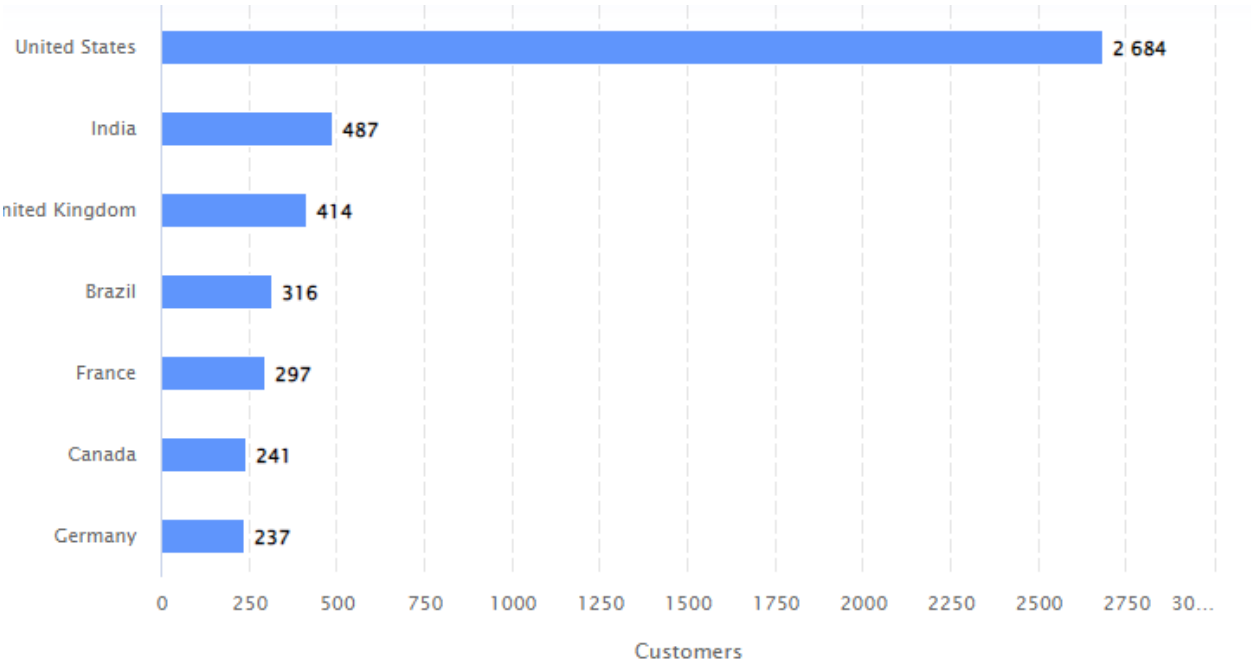
Maven:

Around the globe in 2023, over 7000 enterprises are currently using Apache Maven as a build-automated tool.

- Organizations using Maven for build-automation are majorly from the USA States with 2684 customers, i.e., around 36.45% of customers are from the United States.
- Other top nations using Apache Maven are the United Kingdom and India, with 414(5.62%) and 487(6.61%) customers, respectively.
- The three top industries that use Maven for Build Automation are Big Data (253), Software Development (311), and Machine Learning (207).

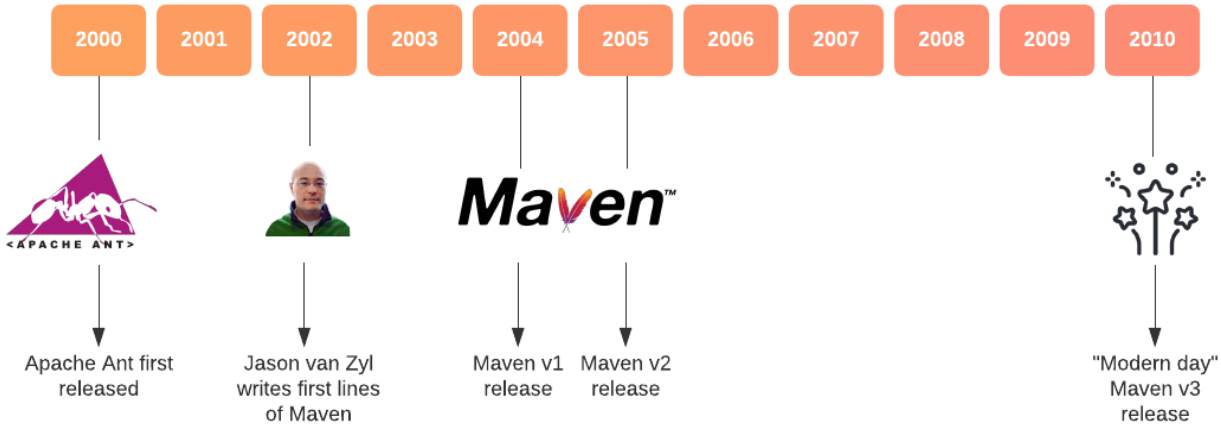


Source

Maven has an enormous community of users contributing to enhancements, communicating use cases, reporting defects, writing documentation, and aiding customers needing assistance. In the case of numerous development teams' environments, Maven can set up and fit the way to function as per standards in the shortest time.

What is Maven?

In the Yiddish language, Maven means “collector/ accumulator of knowledge.” It was initially launched on 13 July 2004.



Source

Maven is one of the popular open-source build tools developed by the Apache Group for building, publishing, and deploying multiple projects for perfect project management. Maven provides software developers build & document the SDLC framework. It is mainly written in Java and is used to build projects written in Scala, C#, Python, Ruby, etc. Based on the POM (Project Object Model), this tool has made the lives of Java developers stress-free while developing reports, scrutinizing builds, and testing automation set-ups.

Maven is an open-source automation and project management tool developed by apache software foundation it is based on POM (Project Object Model).

Maven can build any number of into desired output such as .jar, war, metadata.

Mostly used for Java based projects.

It was initially released on 13 July 2004.

Maven is written in Java.

Maven helps in getting the right jar file for each project as these as these may be different version of separate packages.

To download dependencies it is no more needed to visit the official website of each software it cloud now be easily done by visiting “mvnrepository.com”

Maven in DevOps emphasizes the standardization and simplification of the building procedure, taking care of the following:

- Builds
- Dependencies
- Documentation
- SCMs
- Distribution
- Reports
- Releases

What Maven Does?

1. It makes a project easy to buld.
2. It provides project information (for eg. Log document, cross reference sources, mailing list, dependency list, unit test)
3. Easy to add new dependencies
4. Therefore, apache maven help to manager-
    1. Build
    2. Dependencies
    3. Reports
    4. Releases
    5. Distribution

What is build tool?

A build tool take core of everything from building a process. It does following

- Generate source code
- Generate documentation from source code.
- Compiles source code.
- Install the package code in load repo, server repo or central repository.

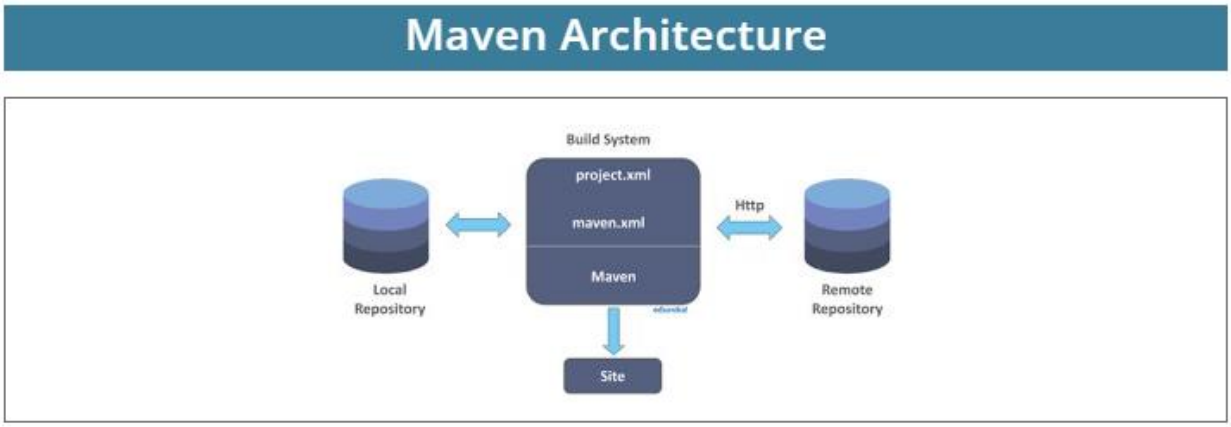
The Need for Maven

Maven helps developers manage Java-based apps through projects that organize and handle code files & build scripts to execute and run compiler tools, version no. for compiled code, plus dependency management that allows one project to reference a version of other projects.

- Build a project using Maven.
- Accumulation and collection of Source Code
- Running Tests (functional tests and unit tests)
- Upload the packages to remote repos (Artifactory, Nexus)
- Packaging the outcomes into WAR's, JAR's, RPM's, etc.

- Using Maven you can add jars & other dependencies of the project effortlessly.
- Maven gives project information (dependency list, unit test reports, log document, etc.)
- Using Maven we can simply integrate our project with a source control system (like Git or Subversion).

Maven Architecture



Source

In the above figure, these are the many components of Maven architecture. This is the local machine's local repository that you function on. This is the central repository and this is the remote web server or the remote repository, so when you specify any dependency in the palm.XML file of Maven. Maven will seek the file in the central repository. If the dependence exists in the central repository Maven will copy that dependence onto your local machine, however, if it isn't present here Maven will get it from the remote repository via the internet, so the internet is mandatory for making use of Maven this is how Maven architecture functions.

The Maven build follows a particular life cycle to deploy & distribute the target project.

There are three build life cycles::

- **default:** the main life cycle as it is responsible for project delivery
- **clean:** to clean the project & remove all files made by the previous build
- **site:** to generate the project's website documentation

Key Features of Maven

Maven is loaded with several useful traits, which goes a long mode towards explaining its popularity. Here are a few of Maven's notable features:

- Easy project setup that follows best practices.
- Simple mode to build projects in which pointless details are hidden.
- Dependency management counting automatic updating.
- Dynamic downloading of crucial Java plug-ins and libraries from Maven repositories.
- **Coherent website of project data** − Using the same metadata under the build procedure, Maven can make a site and a PDF counting complete documentation.
- **Model-based builds** − Maven can build any no. of projects into predefined productivity types such as war, jar, and metadata.
- **Parallel builds** − It examines the project dependency graph and allows you to build schedule modules in parallel.
- **Backward Compatibility** – You can effortlessly port the numerous modules of a project into Maven 3 from old Maven versions.

How can Maven benefit DevOps?

Maven aids the developer in generating a Java-based project more simply. Accessibility of fresh features generated or added in Maven can be simply added to a project in Maven configuration.

It enhances the project performance and building procedure. The core feature of Maven is that it can automatically download the project dependency libraries.

Below are the instances of popular IDEs supporting development with Maven Framework:

- NetBeans
- Eclipse
- MyEclipse
- JBuilder
- IntelliJ IDEA

Processes that can be managed using Maven:

- Building
- Documenting
- Reporting
- Dependencies
- SCMs
- Launches
- Distribution

Problem without Maven:

1. Adding set of jars in each project-

In core of stubs, spring, we need to add jar files in each project it much include all the dependencies of jars also.

2. Creating the right project structure-

We must create the right project structure in srvlet, strubs etc. otherwise it will not be executed

For eg.

.war file layout

3. Building and deploying the project-

We must have to build and deploy the project so that it may work.

POM (Project Object Model)

- POM refers to the XML files that have all the information regarding project and configuration details.
- Main configuration file is pom .xml.
- It has the description of the project details regarding the versioning and configuration management of the project.
- The XML file is in the project home directory.

Pom.xml contains:

- Metadata
- Dependencies
- Kind of project
- Kind of output (.jar, .war)
- Description

Difference between Ant and Maven

Ant and **Maven** both are build tools provided by Apache. The main purpose of these technologies is to ease the build process of a project.

There are many differences between ant and maven that are given below:

Ant	Maven
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.
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.
There is **no life cycle** in Ant.	There is **life cycle** in Maven.
It is **a tool** box.	It is **a framework**.
It is **mainly a build tool**.	It is **mainly a project management tool**.
The ant scripts are **not reusable**.	The maven plugins are **reusable**.
It is **less preferred** than Maven.	It is **more preferred** than Ant.