BuildTools

Build tools are programs that automate the creation of executable applications from source code. Building incorporates compiling, linking and packaging the code into a usable or executable form.

What Does Compiler Mean?
A compiler is a software program that is responsible for changing initial programmed code into a more basic machine language closer to the “bare metal” of the hardware, and more readable by the computer itself. A high-level source code that is written by a developer in a high-level programming language gets translated into a lower-level object code by the compiler, to make the result “digestible” to the processor.

Some of the operations performed by a build tool includes the following:

- downloading dependencies
- compilation of source code into binaries
- packaging
- running tests
- deploying to production environment

Now that you know what a build tool is and what it does, let’s now look at some of the popular build tools in use.

2. Apache Maven

Apache Maven is a build tool of choice for many Java developers. It is based on the concept of POM (Project Object Model) which specifies configuration details and information needed to manage the project. A project using Apache Maven includes a pom.xml file. This file has sections that defines dependencies, plugins etc.

Apache Maven has the benefit of keeping your dependencies up to date and maintaining the whole project configuration in a central place.

3. Gradle

Similar to Maven, Gradle maintains the project configuration details in a central place using a DSL (domain-specific language) called Groovy. Unlike Maven though, it eliminates the XML part of the build script generation.

The Gradle build files are named *build.gradle* and contains a number of task. This tasks include configuring the build and other steps.

The main benefit of Gradle among others is that it is relatively easy to use and provides automatic response for changes in the project source.

4. sbt (Simple Build Tool)

sbt is an open-source build tool generally use for Scala projects. It is also sometimes used for Java projects as well. Some call it Scala Build Tool since it's mainly popular with Scala. But it's actually "simple build tool".

Some of the feature of sbt includes the following:

- provides support for a project with a mix of Java and Scala codes
- natively support compilation of Scala codes and integration of a number of Scala test frameworks
- support for incremental compilation, testing and deployment
- integrates with Scala interpreter

5. Apache Ant

Apache Ant is a popular build tool which originated from the Apache Tomcat project. It was created as a replacement for thee Make build tool for Unix. It is implemented in Java and therefore requires the Java runtime.

Ant uses XML file to describe the build configuration.

Apache Ant provides the following benefits:

- gives you complete control over the build process
- has the flexibility of being applied to any source language
- provide support for project that has a mix of languages including Java, C and C++

6. CMake

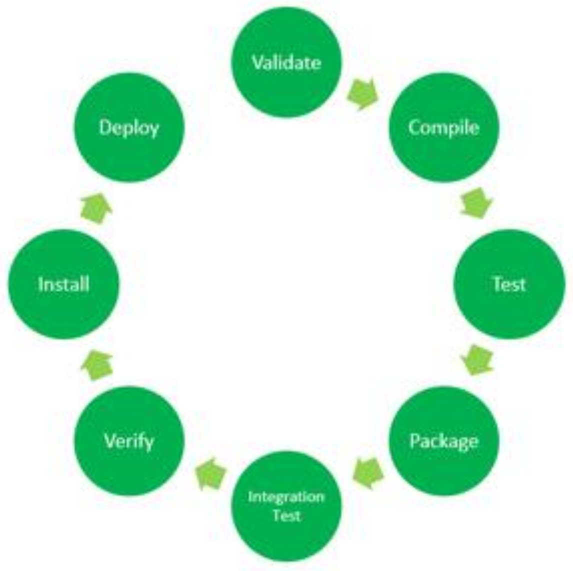
This is an open-source build tool that provides support for managing the build process of software application. It is both cross-platform and free and uses compiler-agnostic methods. CMake however requires a C++ compiler on it's build system.

Some of the benefits of CMake includes:

- provides automatic discovery and configuration
- relatively easy to use
- supports cross-platform discovery of the available system libraries

Maven Lifecycle

Maven is a powerful project management tool that is based on POM (project object model), used for projects build, dependency and documentation. It is a tool that can be used for building and managing any Java-based project. Maven makes the day-to-day work of Java developers easier and helps with the building and running of any Java-based project.



The default Maven lifecycle consists of 8 major steps or phases for compiling, testing, building and installing a given Java project as specified below:

1. **Validate:** This step validates if the project structure is correct. For example – It checks if all the dependencies have been downloaded and are available in the local repository.
2. **Compile:** It compiles the source code, converts the .java files to .class and stores the classes in target/classes folder.
3. **Test:** It runs unit tests for the project.
4. **Package:** This step packages the compiled code in distributable format like JAR or WAR.
5. **Integration test:** It runs the integration tests for the project.
6. **Verify:** This step runs checks to verify that the project is valid and meets the quality standards.
7. **Install:** This step installs the packaged code to the local Maven repository.
8. **Deploy:** It copies the packaged code to the remote repository for sharing it with other developers.

What is a Maven Repository?

In Maven terminology, a repository is a directory where all the project jars, library jar, plugins or any other project specific artifacts are stored and can be used by Maven easily.

Maven repository are of three types. The following illustration will give an idea regarding these three types.

- local
- central
- remote

Local Repository

Maven local repository is a folder location on your machine. It gets created when you run any maven command for the first time.

Central Repository

Maven central repository is repository provided by Maven community. It contains a large number of commonly used libraries.

Remote Repository

Sometimes, Maven does not find a mentioned dependency in central repository as well. It then stops the build process and output error message to console. To prevent such situation, Maven provides concept of **Remote Repository**, which is developer's own custom repository containing required libraries or other project jars.

Maven Dependency Search Sequence

When we execute Maven build commands, Maven starts looking for dependency libraries in the following sequence −

- **Step 1** − Search dependency in local repository, if not found, move to step 2 else perform the further processing.
- **Step 2** − Search dependency in central repository, if not found and remote repository/repositories is/are mentioned then move to step 4. Else it is downloaded to local repository for future reference.
- **Step 3** − If a remote repository has not been mentioned, Maven simply stops the processing and throws error (Unable to find dependency).
- **Step 4** − Search dependency in remote repository or repositories, if found then it is downloaded to local repository for future reference. Otherwise, Maven stops processing and throws error (Unable to find dependency).

The different types of tests

1. Unit tests

Unit tests are very low level and close to the source of an application. They consist in testing individual methods and functions of the classes, components, or modules used by your software. Unit tests are generally quite cheap to automate and can run very quickly by a continuous integration server.

2. Integration tests

Integration tests verify that different modules or services used by your application work well together. For example, it can be testing the interaction with the database or making sure that microservices work together as expected. These types of tests are more expensive to run as they require multiple parts of the application to be up and running.

3. Functional tests

Functional tests focus on the business requirements of an application. They only verify the output of an action and do not check the intermediate states of the system when performing that action.

There is sometimes a confusion between integration tests and functional tests as they both require multiple components to interact with each other. The difference is that an integration test may simply verify that you can query the database while a functional test would expect to get a specific value from the database as defined by the product requirements.

4. End-to-end tests

End-to-end testing replicates a user behavior with the software in a complete application environment. It verifies that various user flows work as expected and can be as simple as loading a web page or logging in or much more complex scenarios verifying email notifications, online payments, etc...

End-to-end tests are very useful, but they're expensive to perform and can be hard to maintain when they're automated. It is recommended to have a few key end-to-end tests and rely more on lower level types of testing (unit and integration tests) to be able to quickly identify breaking changes.

5. Acceptance testing

Acceptance tests are formal tests that verify if a system satisfies business requirements. They require the entire application to be running while testing and focus on replicating user behaviors. But they can also go further and measure the performance of the system and reject changes if certain goals are not met.

6. Performance testing

Performance tests evaluate how a system performs under a particular workload. These tests help to measure the reliability, speed, scalability, and responsiveness of an application. For instance, a performance test can observe response times when executing a high number of requests, or determine how a system behaves with a significant amount of data. It can determine if an application meets performance requirements, locate bottlenecks, measure stability during peak traffic, and more.

7. Smoke testing

Smoke tests are basic tests that check the basic functionality of an application. They are meant to be quick to execute, and their goal is to give you the assurance that the major features of your system are working as expected.

Smoke tests can be useful right after a new build is made to decide whether or not you can run more expensive tests, or right after a deployment to make sure that they application is running properly in the newly deployed environment.

JIRA ticket

A ticket in Jira, or any other service desk platform, is an event that must be investigated or a work item that must be addressed. In Jira Service Desk, tickets entered by customers are called requests. Within a Jira Service Desk queue or in Jira Software, a request is called an issue.

An epic is a large body of work that can be broken down into a number of smaller stories, or sometimes called “Issues” in Jira.

Jira enables teams to plan, track projects, release software, generate reports, and automate processes using a single platform.

CAB

Requesting approval from your Change Advisory Board (CAB) members allows them to review and approve changes to reduce the risk of impacting service quality. You need to be a Jira admin to configure approval from CAB members.

> **Web servers** are the foundation of the internet, providing the infrastructure for websites and applications to be accessed by users. There are several types of web servers available, each with its own advantages and disadvantages. This article will explain the different types of web servers and their features.

Apache Web Server

Apache is the most popular web server in the world, powering over half of all websites. It is an open-source web server, meaning that it is free to use and modify. Apache is highly configurable and can be used to host a variety of websites and applications. It is also highly secure and reliable, making it a great choice for businesses. Apache is available for both Windows and Linux operating systems.

Microsoft IIS Web Server

Microsoft IIS is a web server developed by Microsoft for Windows operating systems. It is a closed-source web server, meaning that it is not open to modification. IIS is a powerful web server that is capable of hosting a variety of websites and applications. It is also highly secure and reliable, making it a great choice for businesses. IIS is available for both Windows and Linux operating systems.

Nginx Web Server

Nginx is an open-source web server developed by a Russian software company. It is a lightweight web server that is capable of handling high traffic volumes. Nginx is highly configurable and can be used to host a variety of websites and applications. It is also highly secure and reliable, making it a great choice for businesses. Nginx is available for both Windows and Linux operating systems.

Node.js Web Server

Node.js is an open-source web server developed by a JavaScript software company. It is a lightweight web server that is capable of handling high traffic volumes. Node.js is highly configurable and can be used to host a variety of websites and applications. It is also highly secure and reliable, making it a great choice for businesses. Node.js is available for both Windows and Linux operating systems.

Tomcat Web Server

Tomcat is an open-source web server developed by an Apache software company. It is a lightweight web server that is capable of handling high traffic volumes. Tomcat is highly configurable and can be used to host a variety of websites and applications. It is also highly secure and reliable, making it a great choice for businesses. Tomcat is available for both Windows and Linux operating systems.

The front end is the part of the website users can see and interact with such as the graphical user interface (GUI) and the command line including the design, navigating menus, texts, images, videos, etc. The backend, on the contrary, is part of the website users cannot see and interact with.
The visual aspects of the website that can be seen and experienced by users are frontend. On the other hand, everything that happens in the background can be attributed to the backend.
Languages used for the front end are HTML, CSS, and JavaScript while those used for the back end include Java, Ruby, Python, and .Net.

The backend is the server side of the website. It stores and arranges data, and also makes sure everything on the client side of the website works fine. It is part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

Back End Languages

- PHP:
- C++:
- Java:

- <u>Python</u>:
- <u>Node.js</u>: