IntelliPaat

Maven™

Agenda

01 Why Maven?

02 What is Maven?

03 What does Maven do?

04 What is a POM file?

05 Maven Life Cycles

06 Maven Repositories

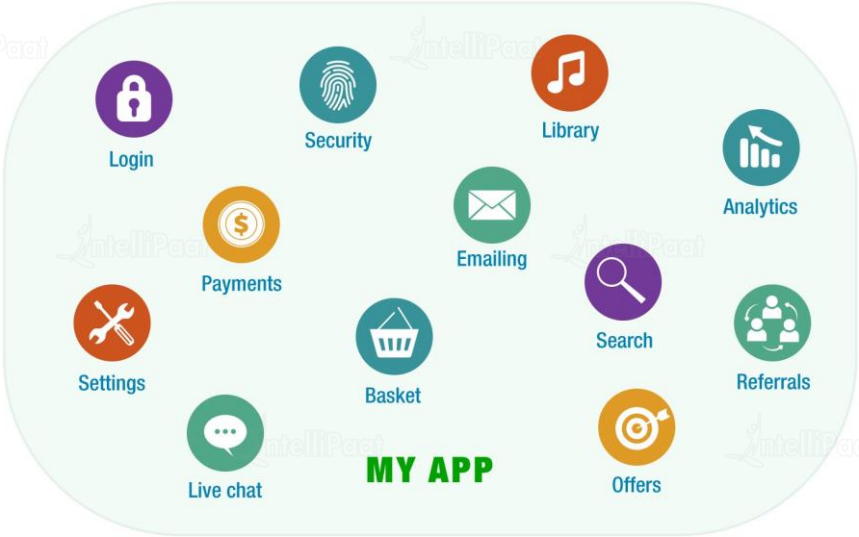
07 Maven Installation

08 Maven Hands-on

Why Maven?

Development of Applications

When an application is written, it is usually divided into smaller modules

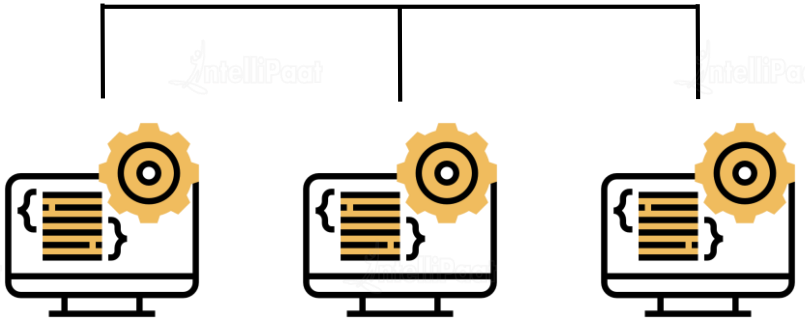


Why do we divide into multiple modules?

It’s easier to go back and check for any bugs or errors if they can be easily located, but inside a singular big code file it would be difficult to do so. It’s much more easier to add and handle features if this big code is divided into multiple small modules

Single Built Code

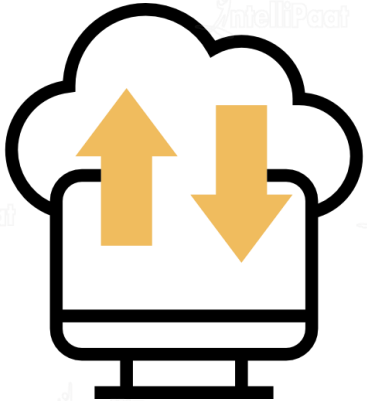
Multiple Modules



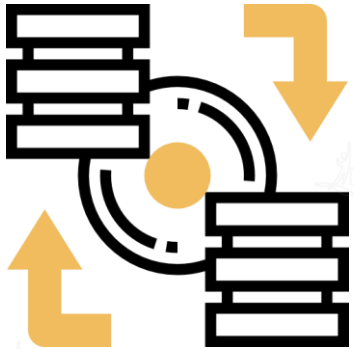
Difficulty with Multiple Modules

While dealing with multiple modules, some problems can emerge as given below:

Manual execution of thousands of modules



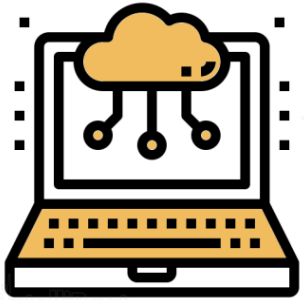
Manual injection of dependencies



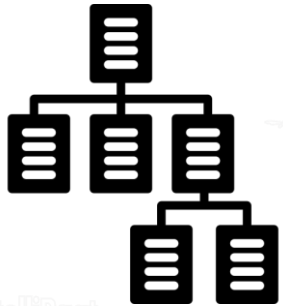
Keeping track of which modules call which modules

Using Maven

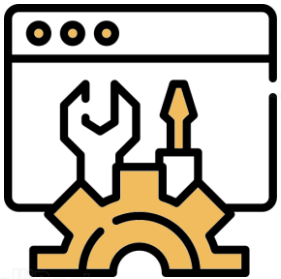
Since Maven is a build tool, it can help in building an application that has several different modules. It helps doing so by:



Downloading dependencies



Enforcing a directory structure

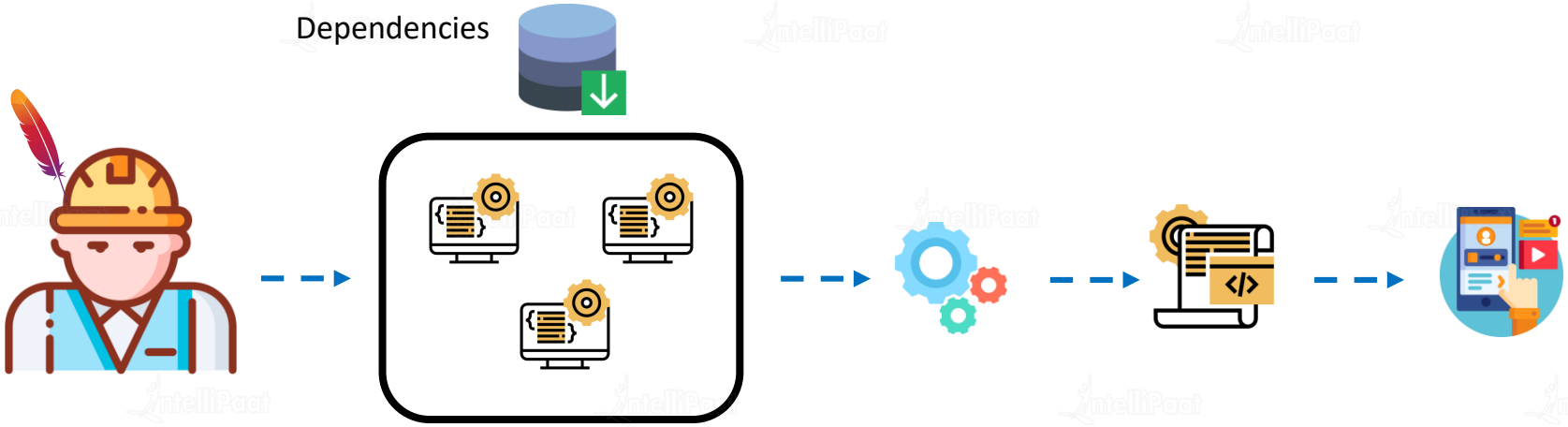


Finally, helping in maintaining an application

What is Maven?

What is Maven?

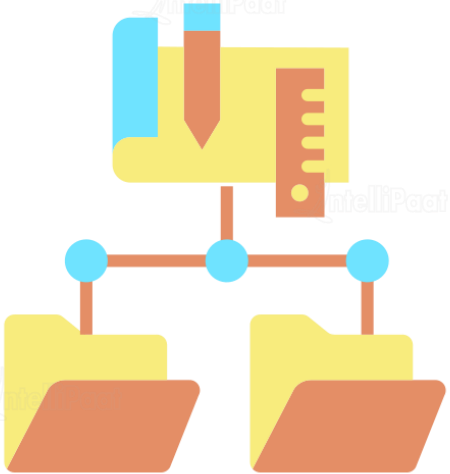
Maven is commonly referred to as a build tool that is used to manage the entire life cycle of a project, generate reports, and store documents with its POM repository



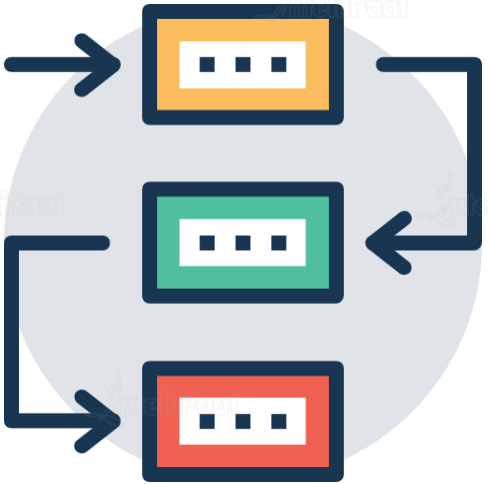
What does Maven do?

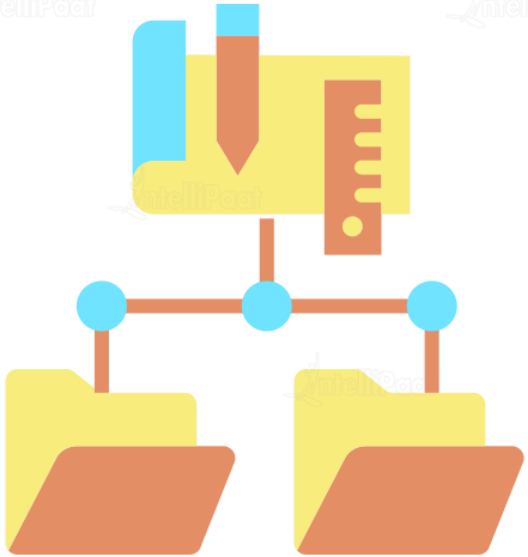
What does Maven do?

Enforce a Directory Structure



Manage and Download Project Dependencies

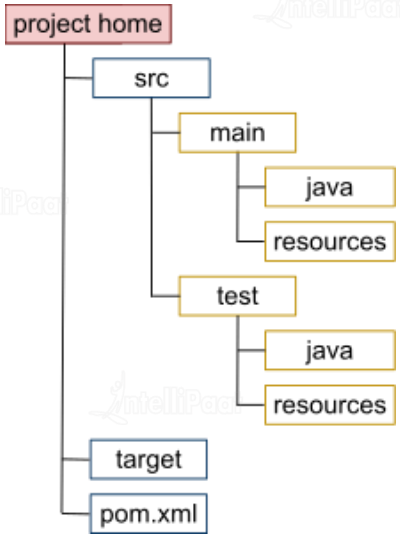




Enforcing a Directory Structure

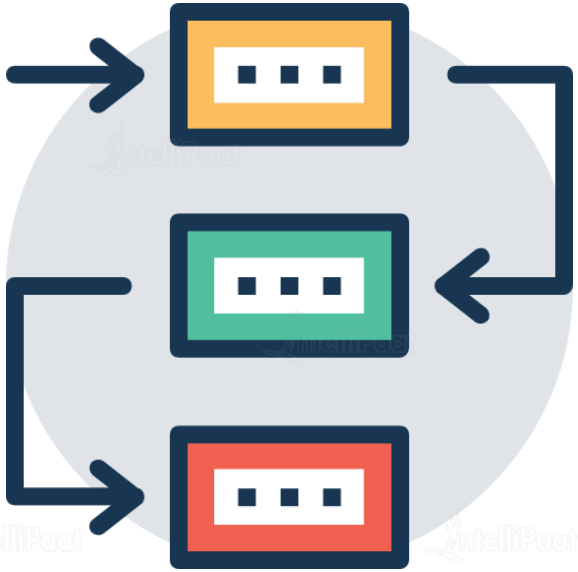
Directory Structure

For Maven to work properly, a standard directory structure needs to be created and be used for storing all the code and it's related resources. Every folder has its own reason for existence



root@ip-172-31-12-200:/opt/apache-maven-3.6.2/bin/sample

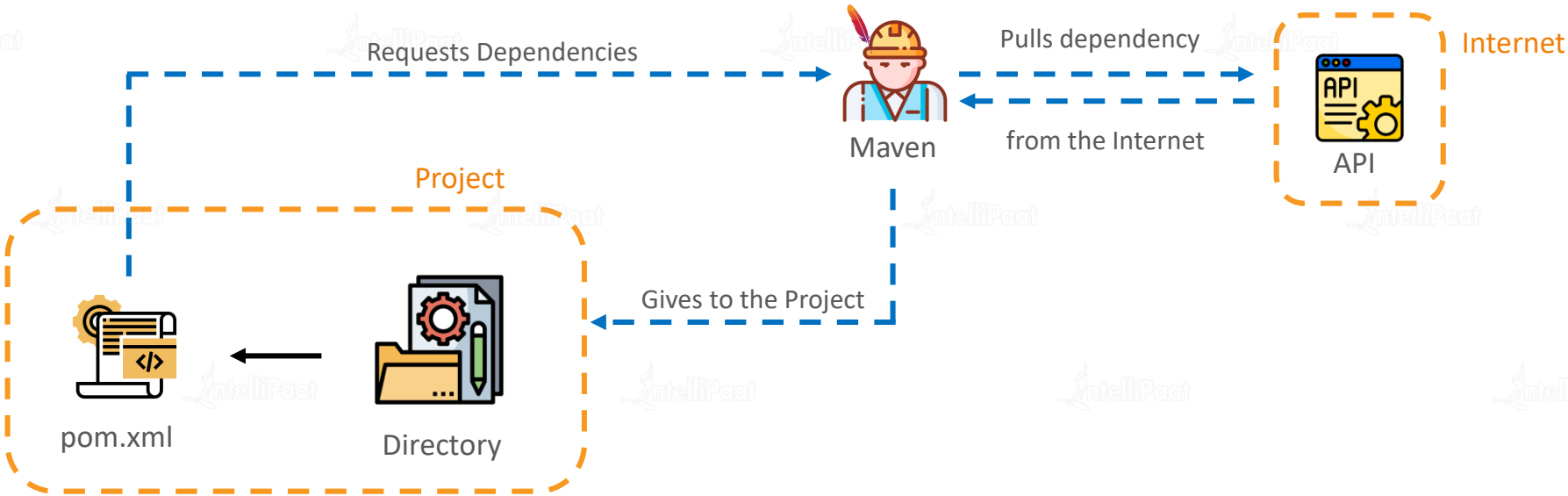
```
[root@ip-172-31-12-200 bin]# cd sample
[root@ip-172-31-12-200 sample]# tree
.
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── general
    │           └── App.java
    └── test
        └── java
            └── general
                └── AppTest.java

7 directories, 3 files
[root@ip-172-31-12-200 sample]#
```



Managing and Downloading Project Dependencies

Project Dependencies

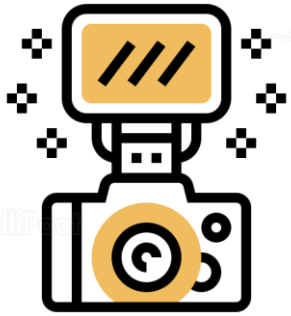
Projects may need Java APIs or frameworks that are packaged in their own JAR files. These JAR files are needed in the class path when a project code is compiled



Project Dependencies

There are two types of dependencies:

External Dependencies

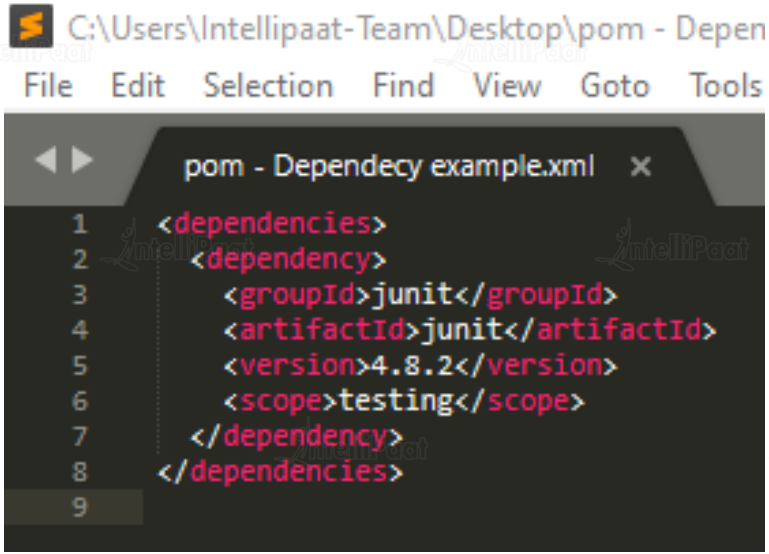


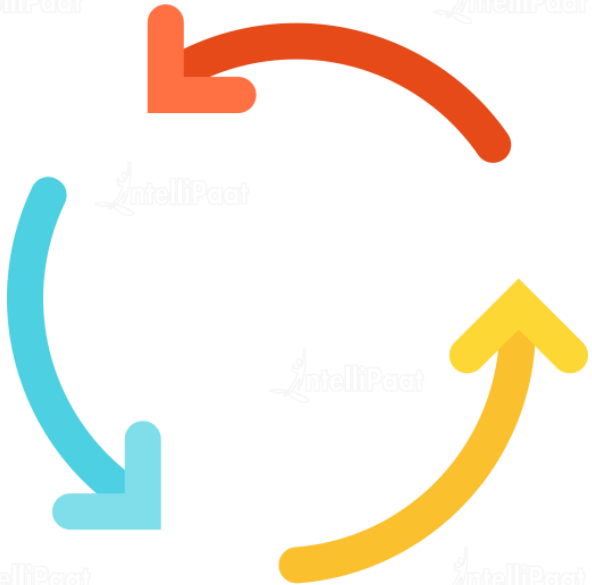
Snapshot Dependencies

Building POM Files

POM (Project Object Model) File

Every Maven project directory needs a **pom.xml** file. These files contain all details necessary for Maven to effectively execute a project. The POM file is stored in 'src' or in the root directory





Maven Life Cycles

Maven Life Cycles

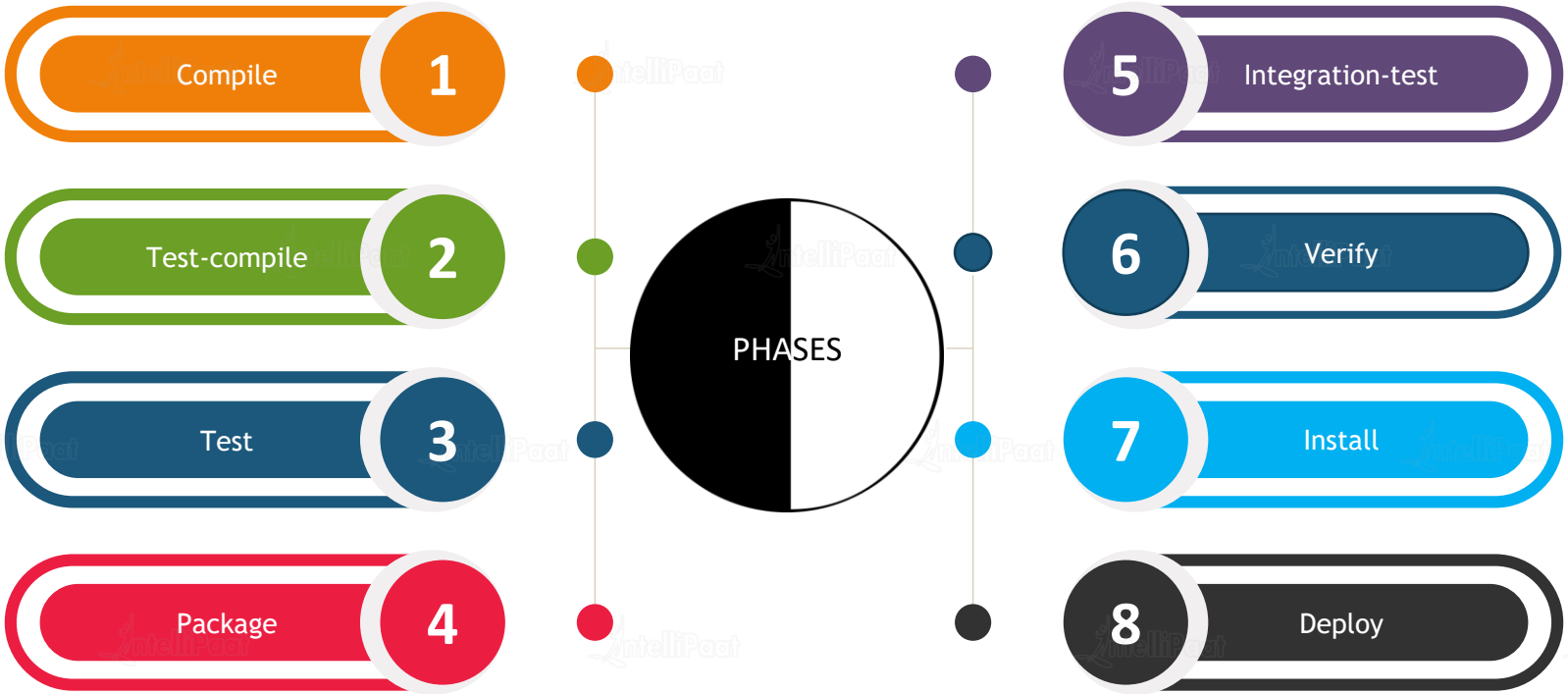
Maven is used to manage the entire life cycle of a project; it generates reports and stores documents with its POM repository. When Maven builds a software, it follows a build cycle. It is of three types:

Default Life Cycle

Clean Life Cycle

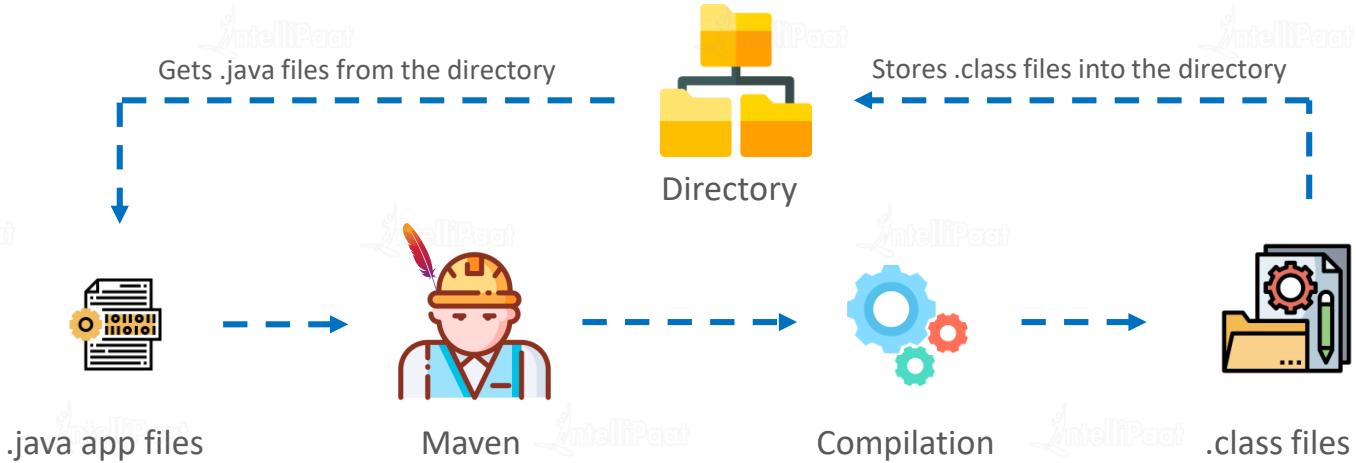
Site Life Cycle

Default Life Cycle



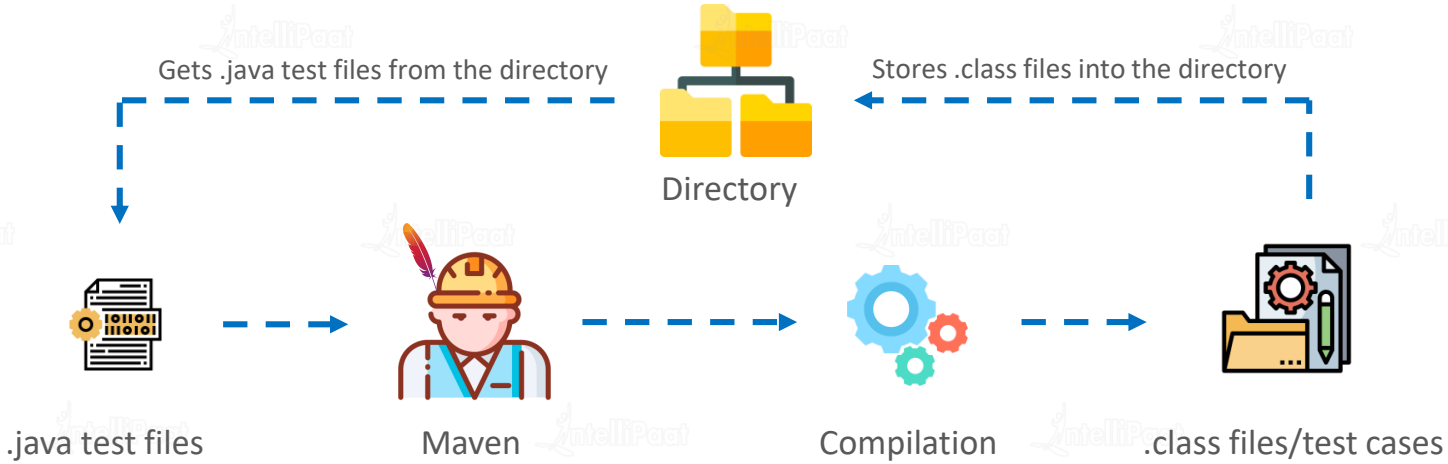
1. Compile Phase

During the compile phase, Maven will compile all **.java** files present in the main directory into **.class** files and put them back into the main directory in the dedicated folder



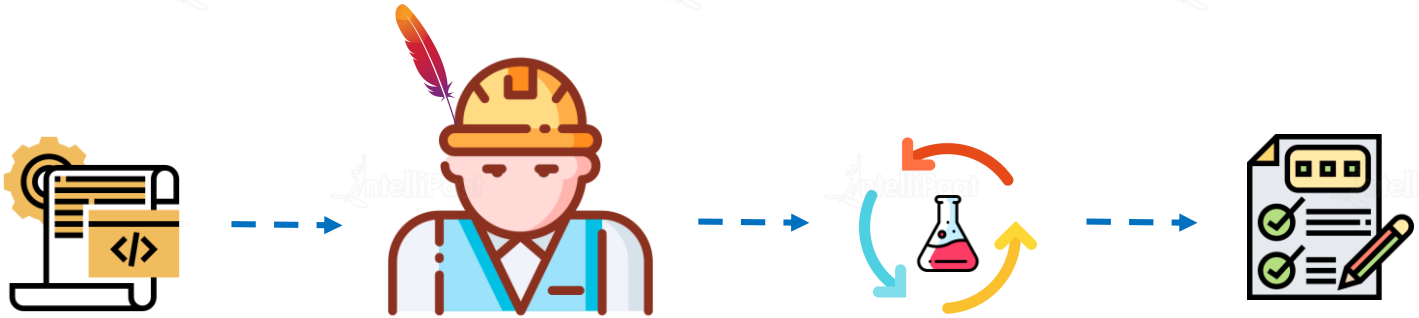
2. Test-compile Phase

During the test-compile phase, Maven will take all **.java** test files from the main directory and compile them into **.class** files that will then be stored back in the main directory in the dedicated folder



3. Test Phase

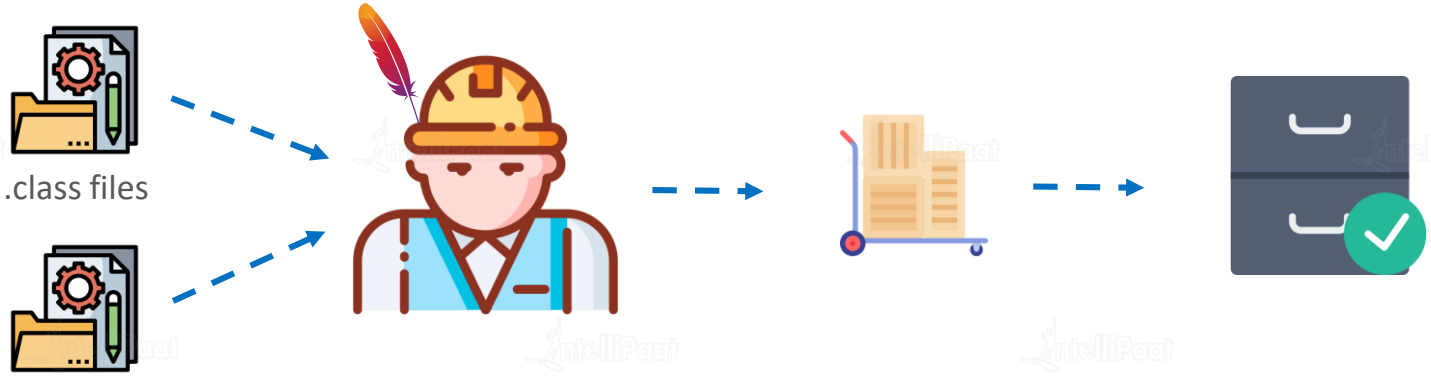
During the test phase, Maven will execute the specified test cases and create a summary log



4. Package Phase

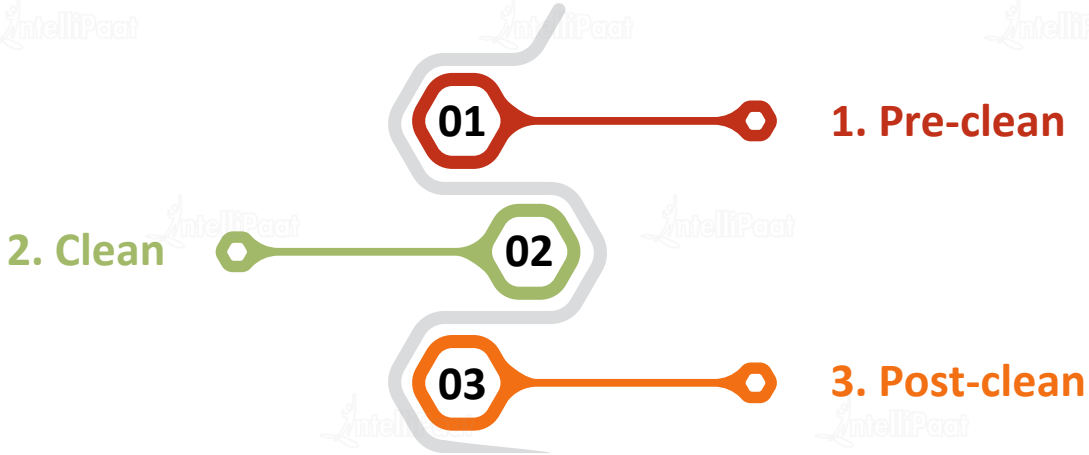
During the package phase, Maven will package all **.class** files and resources into one file. This file will be formatted into one of the three types given below:

JAR Archive | WAR Archive | EAR Archive



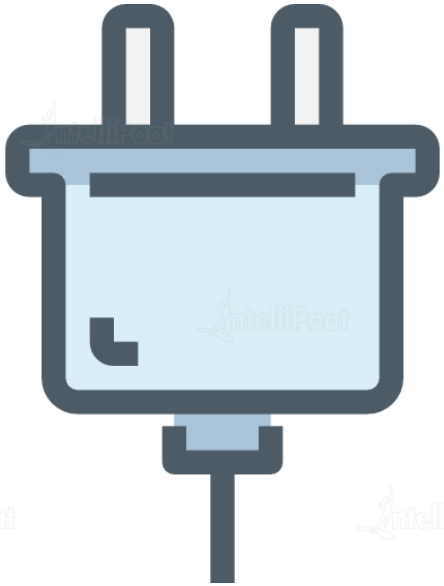
Clean Life Cycle

In this life cycle, all the files that were produced by Maven will be automatically removed. There are three steps in it



Site Life Cycle

The site life cycle generates a project file documentation in the HTML format. However, this cycle is usually ignored as its output is not that useful. There are four stages in it

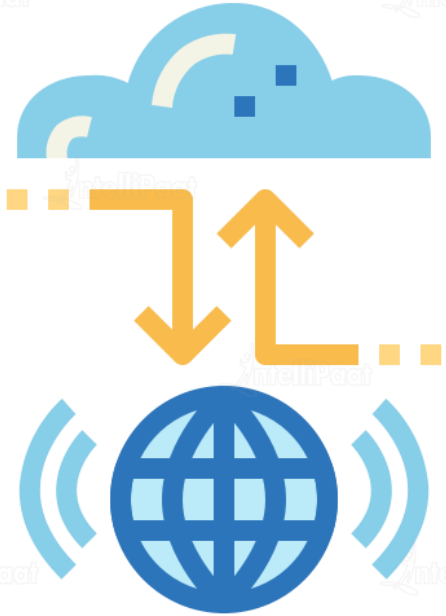


Maven Plugins

3. Maven Plugins

Plugin is a file that contains the knowledge of what is supposed to happen when a phase is executed

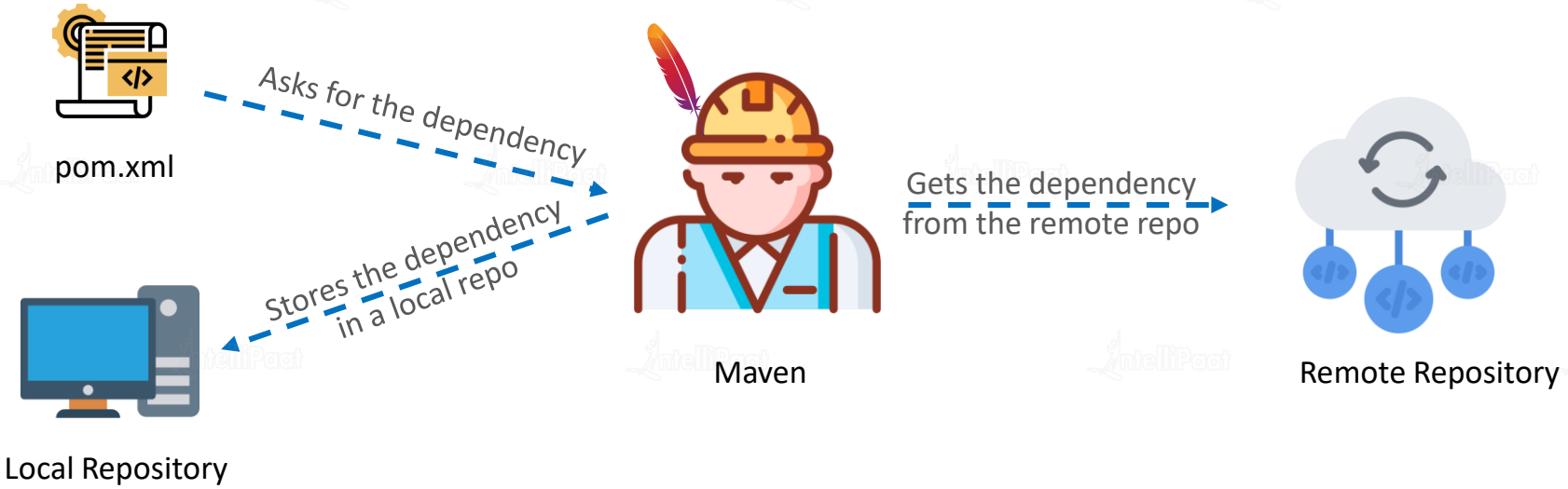
Phase	Maven Plugin	Goal
Clean	Clean Plugin	Clean
Site	Site Plugin	Site
Process-resources	Resources Plugin	Resource
Compile	Compile Plugin	Compile
Test	Surefire Plugin	Test
Package	Varies based on the packaging [JAR/WAR/EAR]	Jar (in case of JAR packaging)
Install	Install Plugin	Install
Deploy	Deploy Plugin	Deploy



Maven Repositories

Maven Repositories

After Maven understands which all dependencies are needed from the pom.xml file, it will download those dependencies from remote repositories and then store them in the local repository for current or future use



Maven Repositories

Local Repository

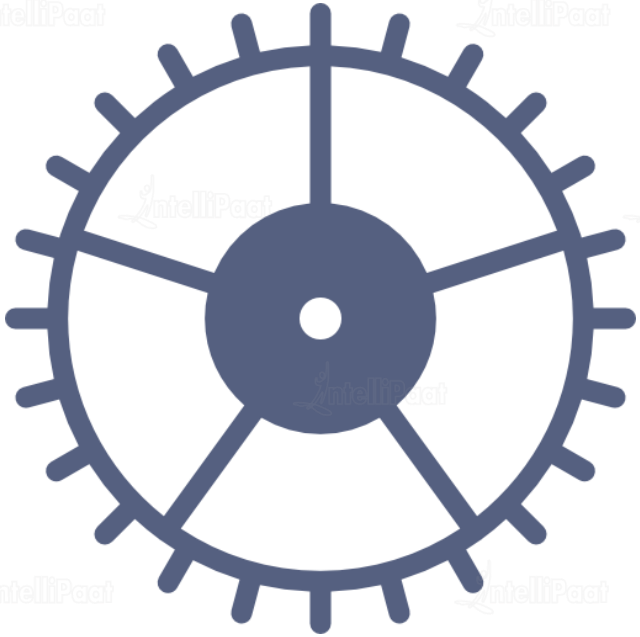
It’s the repository that is stored in the Developer’s system

Central Repository

It’s the repository that is maintained by the Maven community

Remote Repository

It’s the type of repository which is used by companies to maintain integral projects



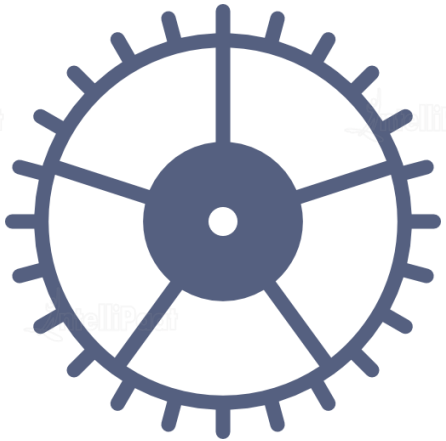
Maven Installation

Maven Installation

Java is a dependency for Maven

- Install Java 1.8.0 using the package installer

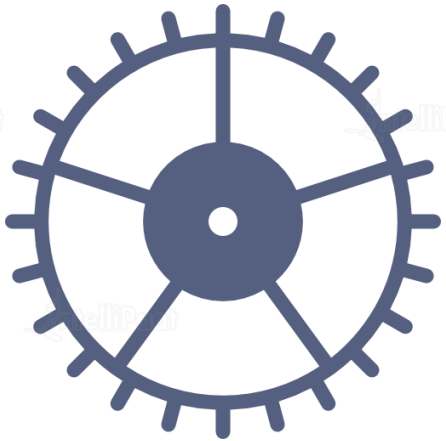
yum install java-1.8.0-openjdk-devel



Maven Installation

- Get the latest version of Apache directly from its website (copy the link address)

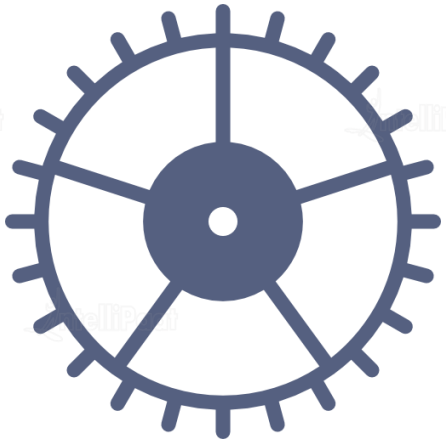
http://mirrors.estointernet.in/apache/maven/maven-3/3.6.2/binaries/apache-maven-3.6.2-bin.tar.gz



Maven Installation

- Unzip the .tar file into/opt the directory

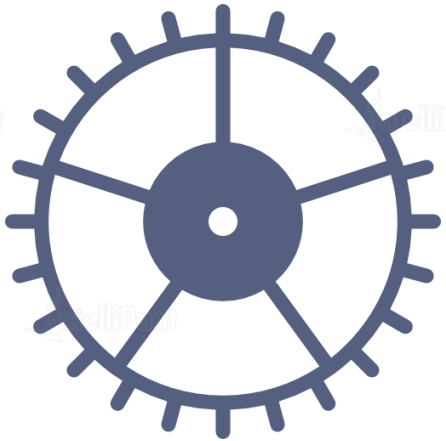
tar zxf apache-maven-3.6.2-bin.tar.gz



Maven Installation

- Go into the unzipped folder and then into the bin folder

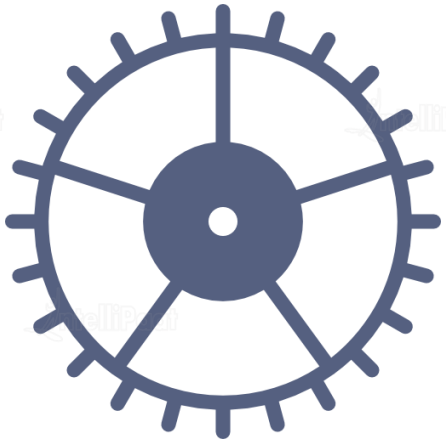
cd /apache-maven-3.6.2/bin

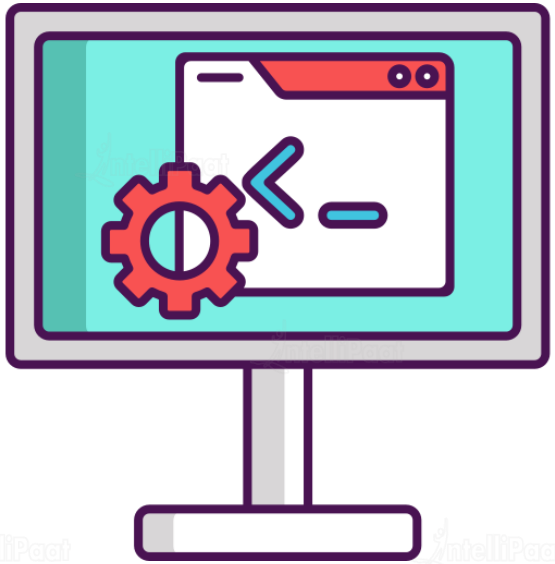


Maven Installation

- Set the environment variable so that the package can be accessed from all over the system

export PATH=$PATH:/opt/apache-maven-3.6.2/bin



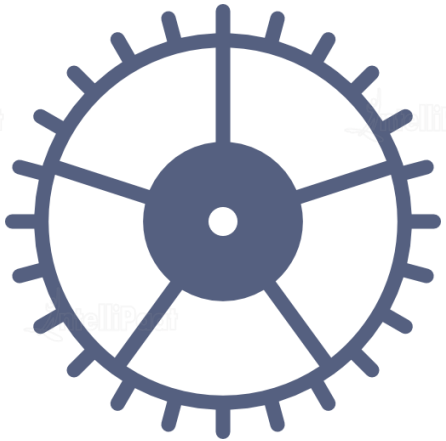


Maven Hands-on

Building a Package

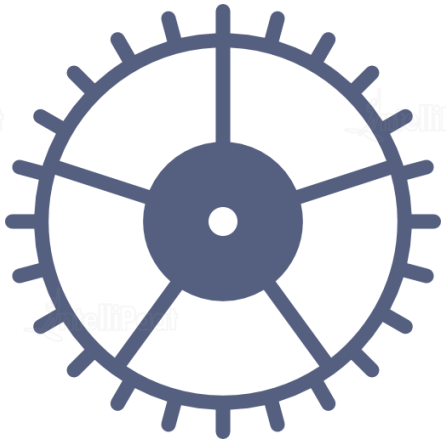
- Create a directory structure for:

mvn archetype:generate



Building a Package

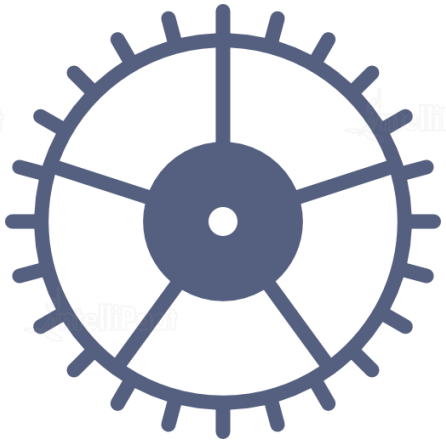
- Maven will prompt you for various details regarding the project



Building a Package

- For better visualization, install tree

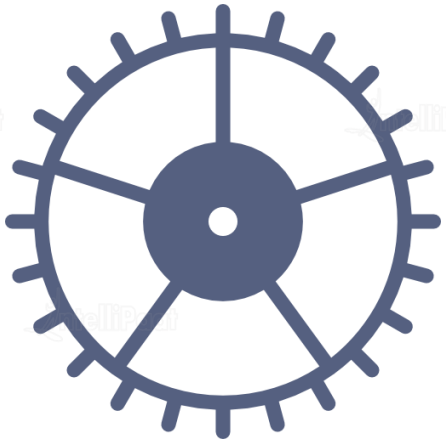
Yum install tree



Building a Package

- Compile the java files

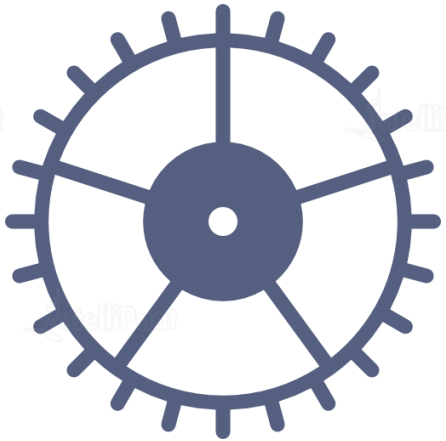
mvn compile



Building a Package

- Run the test cases on the project

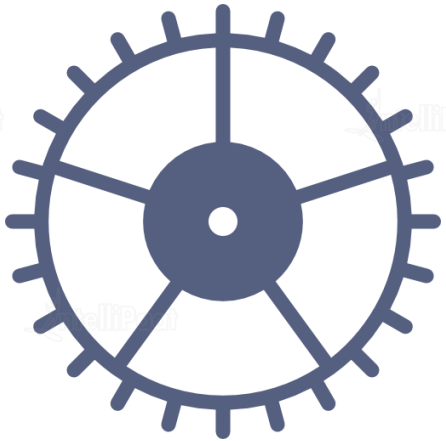
mvn test



Building a Package

- Package the project into one file

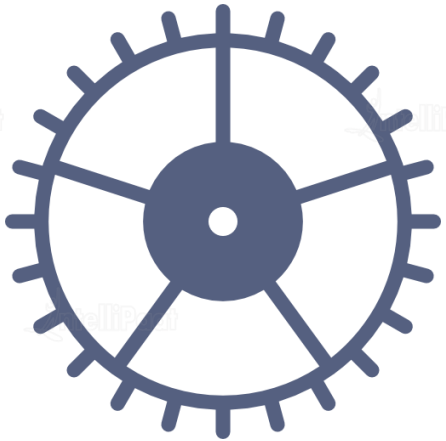
mvn package



Building a Package

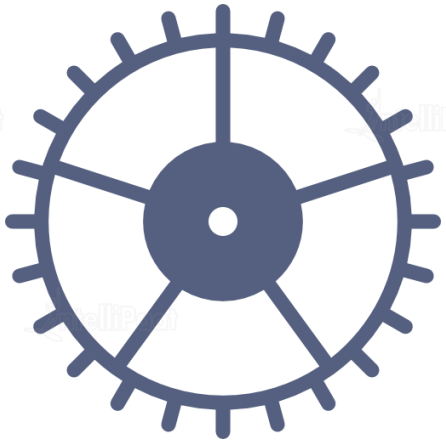
- Install Git and download the sample project from Intellipaat repo

https://github.com/gkdevops/PetClinic.git



Building a Package

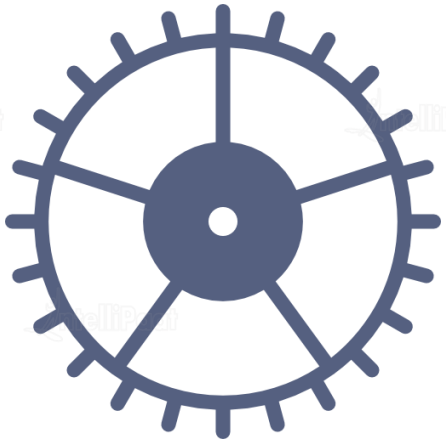
- Package the sample project (Maven will automatically compile and test it before packaging it)



Building a Package

- Now, to execute the file to see if the sample project works or not, make use of Apache Tomcat (Go to its website and copy the download link)

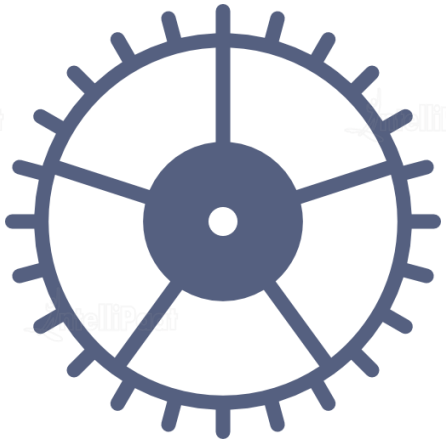
http://mirrors.estointernet.in/apache/tomcat/tomcat-9/v9.0.24/bin/apache-tomcat-9.0.24.tar.gz



Building a Package

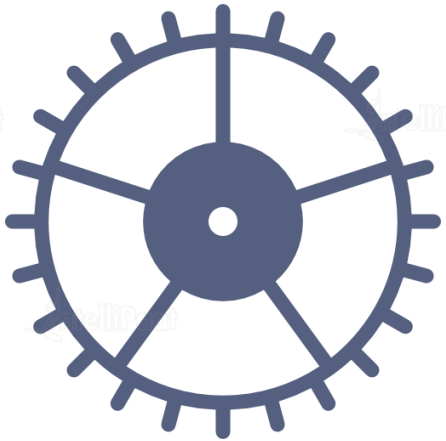
- To install, follow the same procedure as for Maven. Once installed, run the following:

./startup.sh



Building a Package

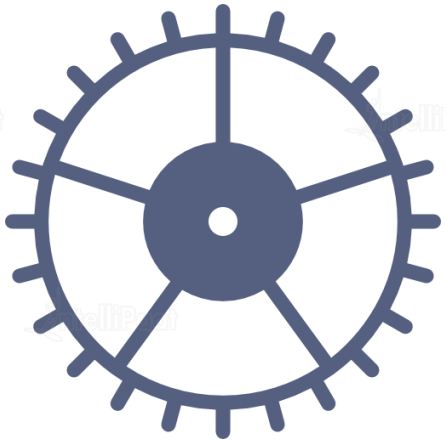
- Tomcat works on Port 8080. To check if the installation was successful, go to **<IP address>:8080** on the browser



Building a Package

- To copy the build JAR file, you need to copy it to Tomcat's directory

cp petclinic.war /opt/apache-tomcat-9.0.24/webapps/

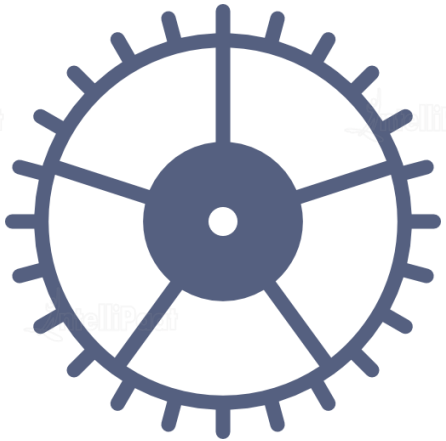


Building a Package

- Now, to deploy the sample project, restart Tomcat

./shutdown.sh

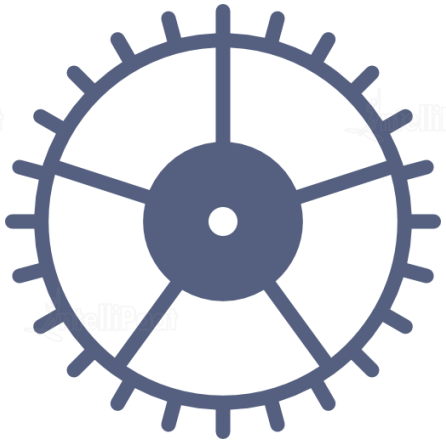
./startup.sh



Building a Package

- Go to the browser and check out:

<IP address>:8080/<Sample Project>



India: +91-7847955955

US: 1-800-216-8930 (TOLL FREE)

sales@intellipaat.com

24/7 Chat with Our Course Advisor