

## I. Configure Maven in Jenkins

1. Go to Jenkins Dashboard -> Manage Jenkins -> Manage plugins -> Available -> Maven Integration -> Install
2. Go to Manage Jenkins->Global tool configuration->Maven -> Add Maven\_home variable value (i.e. path of the maven file on your system).
3. Go to Jenkins Dashboard -> New Item -> Maven Project option will be available

## II. Build a Maven project

1. Go to Jenkins Dashboard -> New Item -> Choose name for the Maven Project (e.g. MyFirstMavenExample)

Enter an item name

MyFirstMavenExample

» A job already exists with the name 'MyFirstMavenExample'

- Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Maven project**  
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- External Job**  
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
- Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

GitHub Organization

2. On Configure Page, set the following:

a). Discard Old builds

Days to keep builds: 1

Max of builds to keep: 5

The screenshot shows the 'General' tab of a Maven build configuration interface. The 'Maven project name' is 'MyFirstMavenExample'. The 'Description' field is empty. The 'Discard old builds' checkbox is checked. The 'Strategy' is set to 'Log Rotation'. The 'Days to keep builds' is set to '1', with a note: 'if not empty, build records are only kept up to this number of days'. The 'Max # of builds to keep' is set to '5', with a note: 'if not empty, only up to this number of build records are kept'. There is an 'Advanced...' button. At the bottom, there is a 'GitHub project' checkbox and a 'HTML5 Notification Configuration' section with 'Save' and 'Apply' buttons.

b). JDK (to be used for this project) Java-1.8

c). Build -> advanced -> enable the following

- Resolve Dependencies during Pom parsing
- Use custom workspace (add path of the folder containing pom.xml)
- Goals & Options = Clean Compile Test

**Build**

Root POM:

Goals and options:

MAVEN\_OPTS:

☐ Incremental build - only build changed modules

☐ Disable automatic artifact archiving

☐ Disable automatic site documentation artifact archiving

☐ Disable automatic fingerprinting of consumed and produced artifacts

☒ Enable triggering of downstream projects

☒ Block downstream trigger when building

☐ Build modules in parallel

☐ Use private Maven repository

☒ Resolve Dependencies during Pom parsing

☐ Run Headless

☐ Process Plugins during Pom parsing

☒ Use custom workspace

Directory:

Validation Level:

Settings file:

### 3). Apply & Save

### 4). Go to Project Window and Click 'Build Now'.

\*Error : Compilation error - No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?

Probable cause: not able to connect to/find jdk.

Solution:

1. Go to Jenkins dashboard -> Manage Jenkins -> Configure System -> Global properties -> Add Environment Variable : JAVA\_HOME and value
2. Also check in the system of your windows : System -> advanced settings -> environment variables Set the PATH : Append with the folder path of the jdk If, user variable is pointing towards JRE, then update it to point towards java path.