**⚡ Difference Between Jenkins and Maven**

| **Feature** | **Jenkins** | **Maven** |
| --- | --- | --- |
| **Type** | Automation/CI-CD tool | Build automation tool |
| **Purpose** | Orchestrates the whole software pipeline (build → test → deploy → monitor). | Builds the project: compiles, tests, packages, manages dependencies. |
| **Written in** | Java (web application) | Java (command-line tool) |
| **Core Function** | Runs jobs/pipelines when triggered (push to GitHub, schedule, etc.). | Executes build lifecycle defined in pom.xml. |
| **Input** | Jenkinsfile (pipeline script) or job configuration. | pom.xml (Project Object Model file). |
| **Scope** | Big picture (end-to-end automation). | Specific to building and managing Java projects. |
| **Example Command** | Trigger job → mvn clean install | mvn clean install builds project |
| **Plugins/Integration** | Integrates with GitHub, Docker, Kubernetes, Ansible, Maven, etc. | Integrates mainly with repositories (Maven Central, Nexus, Artifactory). |
| **When Used** | When you need CI/CD automation. | When you need to build and manage Java projects. |

**✅ Simple Example (Plain Words)**

* **Maven alone**:  
  You run mvn package → it compiles your Java code, runs tests, and produces a .jar or .war.
* **Jenkins with Maven**:  
  Jenkins job triggers automatically when code is pushed to GitHub → Jenkins tells Maven to run mvn clean install → then Jenkins deploys the .jar/.war to a server.

👉 You can think of it like:

* **Maven = the builder** (construction worker 🧱).
* **Jenkins = the project manager** (assigns work, coordinates, makes sure the house is built and delivered 🏗️).