Maven –

Apache Maven is a software project management and build management tool for Java Frameworks.

1. Why Maven?

* Central repository to get dependencies
* Maintaining common structure across the organization
* Flexibility in Integrating with CI tools
* Plugins for Test framework execution.

1. Install Maven
2. Set System variables to recognize Maven
3. Understanding Maven terminologies

**Artifact**: An artifact is a file, usually a JAR, that gets deployed to a Maven repository.

**GroupId**: groupId will identify your project uniquely across all projects,

**archetype:generate** ; Generates a new project from an archetype

1. Creating Maven Project

mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

1. Integrate Maven with Eclipse
2. Maven Phases for Test Automation
3. Clean, compile, test

Text

Description automatically generated

1. Understanding POM.xml file dependencies..
2. How to find the dependencies of the Softwares?
3. Importance of Maven Sure fire plugin
4. Running Tests with Surefireplugin
5. Integrating Testng into Maven
6. Testng xml files configuration in POM file
7. Setting up Profiles in POM.xml file
8. Running selected Tests only with Maven commands
9. Maven Sure fire reports

Questions-

1. Maven folder structure –

* Src/main/java – java source code for artifact (to maintain source code)
* Src/main/resources - configuration files like per-environment configuration files, and XML configurations
* Src/test/java – java source code for tests (to maintain unit testing code)
* Src/test/resources - configuration files and others used by tests
* Pom.xml – defines dependencies and modules needed during the build lifecycle of a Maven project
* Src
* Target -

Jenkins

1. Importance of Jenkins
2. Jenkins installation
3. Config global settings In Jenkins
4. Understanding Jenkins workspace
5. Configure Jenkins job parameters
6. Post build action plugins
7. TestNG Jenkins plugin to generate reports
8. Scheduling Jenkins builds

* What is Jenkins
* **Jenkins** is an open-source Continuous Integration tool
* **The main aim is to** achieve the Continuous Integration process in an automated fashion.
* Jenkins is used to build and test your software projects continuously by integrating with a testing and deployment technologies.
* Role of Jenkins
* Jenkins is often used for **building projects**; running tests to spot bugs, to analyze static code, and deployment. It also executes repetitive tasks, saves time, and optimizes developing processes
* What is Jenkins used for in testing?
* Jenkins is a popular CI tool.
* It provides numerous plugins for integration with multiple test automation tools and frameworks into the test pipeline.
* When it comes to test automation, Jenkins provides **plugins that help run test suites, gather and dashboard results, and provide details on failures**
* What is continuous integration and how to achieve using Jenkins
* <https://www.edureka.co/blog/what-is-jenkins/>

**Advantages of Jenkins include:**

* It is an open-source tool with great community support.
* It is easy to install.
* It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share it with the community.
* It is free of cost.
* It is built with Java and hence, it is portable to all the major platforms.
* Common test/use cases Jenkins used for

<https://www.quora.com/What-are-some-cool-use-cases-of-Jenkins-other-than-continuous-integration>

* Jenkins pipeline

<https://www.edureka.co/blog/jenkins-pipeline-tutorial-continuous-delivery>

* what is Jenkins job
* how to store Jenkins credentials securely
* different ways to trigger Jenkins job