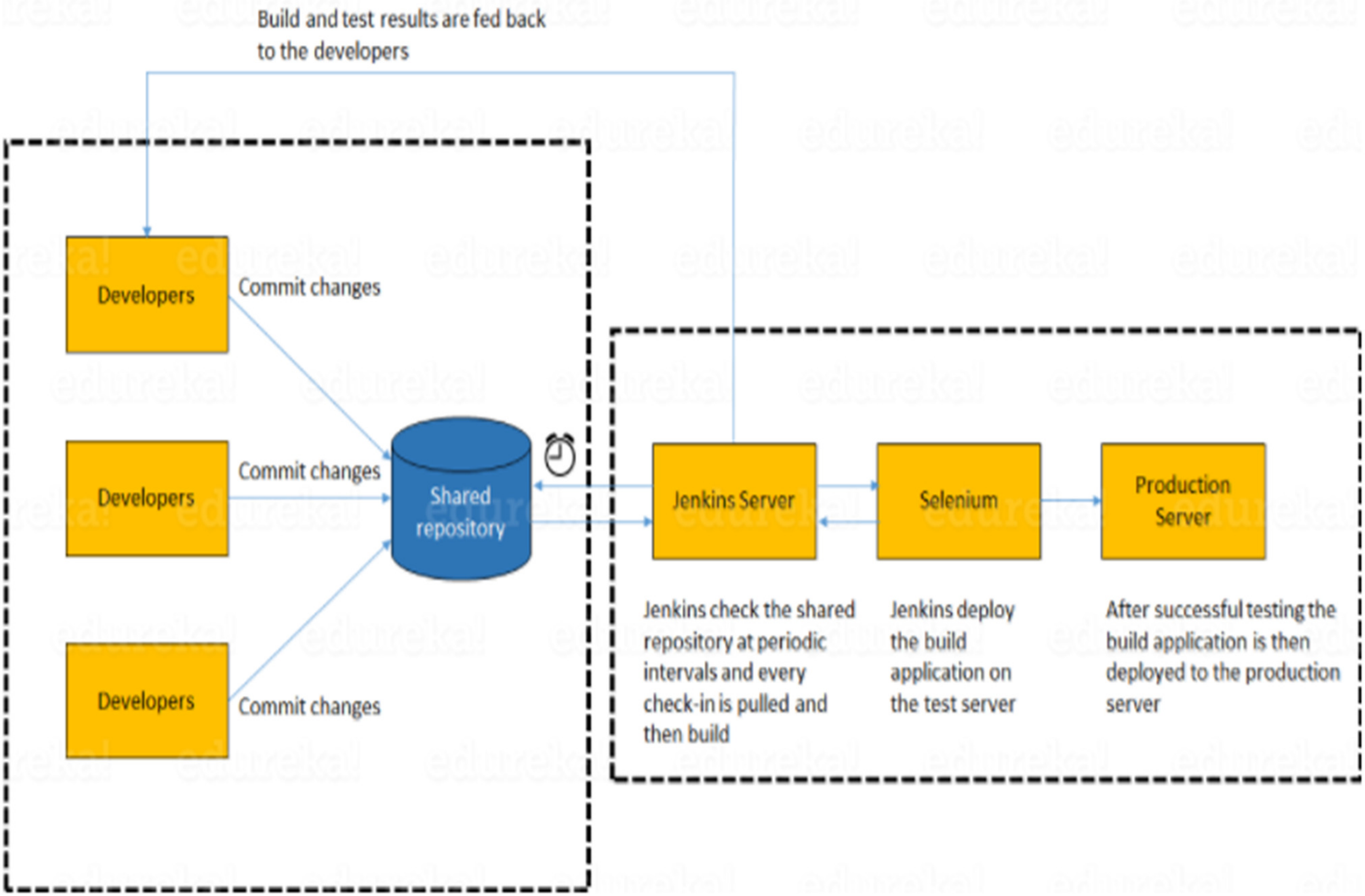
JENKINS INTERVIEW QUESTIONS AND ANSWERS

What is meant by Continuous Integration?

- I will advise you to begin this answer by giving a small definition of Continuous Integration (CI).
  It is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.
  I suggest that you explain how you have implemented it in your previous job. You can refer the below given example:


Build and test results are fed back to the developers
Developers
Commit changes
Developers
Commit changes
Developers
Commit changes
Shared repository
Jenkins Server
Selenium
Production Server
Jenkins check the shared repository at periodic intervals and every check-in is pulled and then build
Jenkins deploy the build application on the test server
After successful testing the build application is then deployed to the production server

- In the diagram shown above:
- Developers check out code into their private workspaces.
- When they are done with it they commit the changes to the shared repository (Version Control Repository).
- The CI server monitors the repository and checks out changes when they occur.
- The CI server then pulls these changes and builds the system and also runs unit and integration tests.
- The CI server will now inform the team of the successful build.
- If the build or tests fails, the CI server will alert the team.
- The team will try to fix the issue at the earliest opportunity.
- This process keeps on repeating.
- Continuous Integration (CI) is a software development practice, where the team members will continuously integrate their work into a shared repository, at least daily – resulting to multiple integrations on a daily basis.

What is Continuous Integration in Jenkins?

- In software development, multiple developers or teams work on different segments of same web application so you have to perform integration test by integrating all modules. In order to do that an automated process for each piece of code is performed on daily bases so that all your codes get tested. This process is known as continuous integration.

- **Why do you need a Continuous Integration of Dev & Testing?**
- For this answer, you should focus on the need of Continuous Integration. My suggestion would be to mention the below explanation in your answer: Continuous Integration of Dev and Testing improves the quality of software, and reduces the time taken to deliver it, by replacing the traditional practice of testing after completing all development. It allows Dev team to easily detect and locate problems early because developers need to integrate code into a shared repository several times a day (more frequently). Each check-in is then automatically tested.

What is the function of CI (Continuous Integration) server ?

- CI server function is to continuously integrate all changes being made and committed to repository by different developers and check for compile errors. It needs to build code several times a day, preferably after every commit so it can detect which commit made the breakage if the breakage happens.
- Note: Other available and popular CI tools are Jenkins, TeamCity, CircleCI , Hudson, Buildbot etc

- **What are the benefits of Continuous Integration (CI)?**
- Following are the key benefits of CI.
- Higher code quality
- Reduced integration risk
- Code in version control works
- Less time for deployment
- Increased confidence and morale

Define the practices that make up effective CI.

- The key practices that make the things smoother for CI are:
- Maintain a single repository
- Automate the build
- Make the build self-testing
- Everyone commits to the mainline every day
- Every commit should build the mainline on an integration machine
- Fix the broken build instantly
- Keep the build fast
- Make it easy for everyone to get the latest executable
- Automate deployment

• What are the principles of "Continuous Integration“?

Maintain a code repository :
Code should be maintained in a Version Control such as CVS,SVN,VSS,ClearCase etc., which could allow multiple developers to work collaboratively in parallel by Versioning the Files (code).

- **Automate the build :**
  Should Have scheduled automated builds by using the tools like "Hudson,TeamCity, CruiseControl" etc.,
  which could automatically checkout(Get) the code from Code Repository and build.
- **Make the build self-testing :**
  During the build process, after compiling the code we can make the Code Self-Testing by executing the Unit Test cases such as JUNIT or Cactus or EasyMock etc. and confirms that nothings is broken.
- **Automate deployment :**
  Once build and self-testing process is done, we should have automated Deployments.
- Tools such as "Hudson,TeamCity, CruiseControl" do complete "Continuous Integration" for you,but you have to have some build script such as ANT or MAVEN scripts to perform some of these Tasks.
- Here is the sequence of tasks that could be done by above tools.
  i) Having Scheduled Builds. (Daily or Hours)
  i) Checkout the code.
  ii) Compile the Code.
  iii) Running Unit Test-Casess.
  iv) FTP code to different hosts

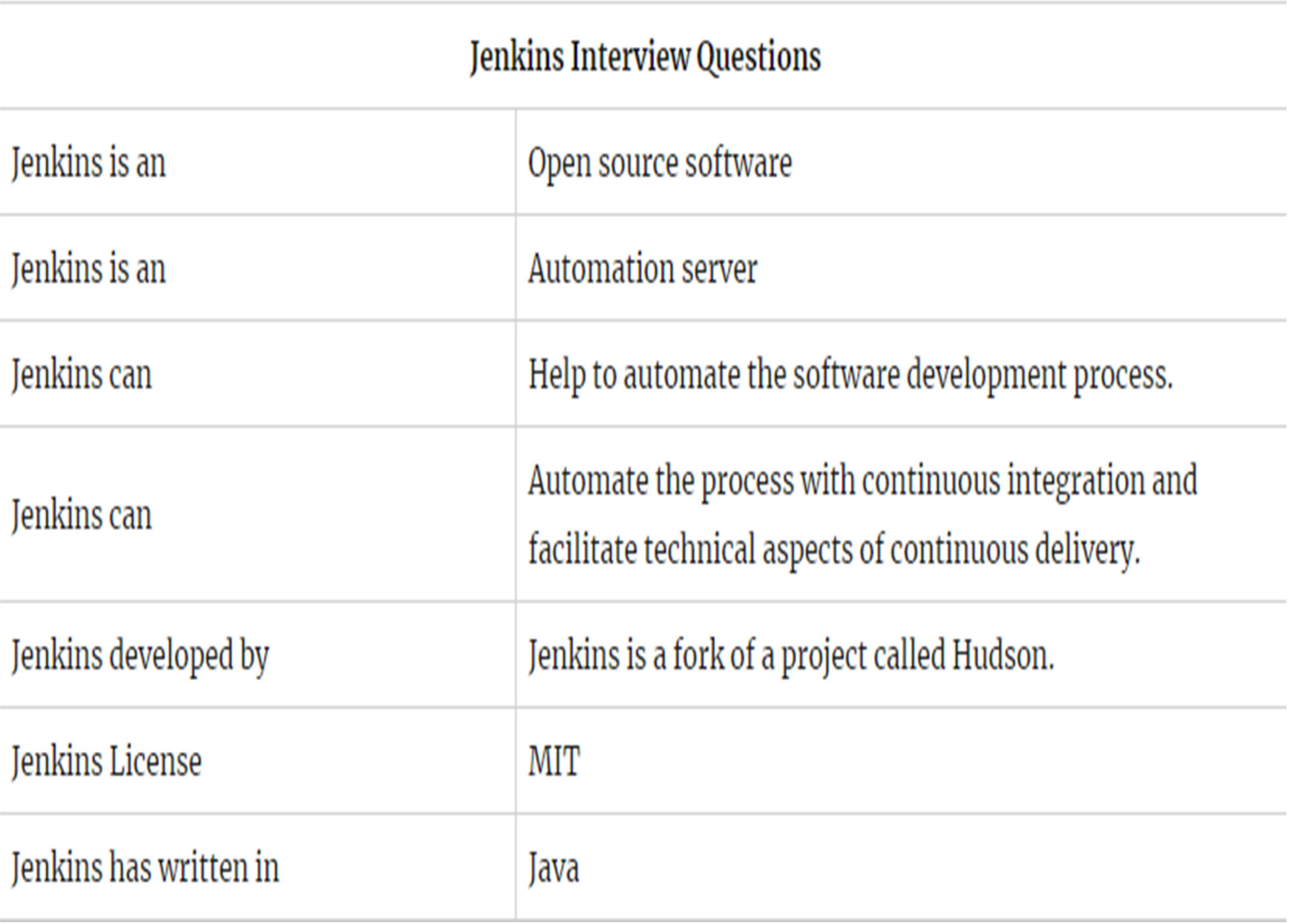
- **Why is Continuous Integration important?**
- Two important reasons:
- Defects found early cost less to fix : When a defect is found immediately after a developer codes it, it takes 10x times less time to fix it compared to finding the defect a month later.
- Reduced Time to Market : Software is always tested. So, it is always ready to move to further environments.

- **How is Continuous Integration Implemented?**
- Different tools for supporting Continuous Integration are Hudson, Jenkins and Bamboo. Jenkins is the most popular one currently. They provide integration with various version control systems and build tools.

- **What are the success factors for Continuous Integration?**
- Implementing the tools for Continuous Integration is the easy part. Making best use of Continuous Integration is the complex bit. Are you making the best use of your continuous integration setup? Here are the things you would need to consider.
- How often is code committed? If code is committed once a day or week, the CI setup is under utilised. Defeats the purpose of CI.
- How is a failure treated? Is immediate action taken? Does failures promote fun in the team?
- What steps are in continuous integration? More steps in continuous integration means more stability.
  - Compilation
  - Unit Tests
  - Code Quality Gates
  - Integration Tests
  - Deployment
  - Chain Tests
- More steps in continuous integration might make it take more time but results in more stable application. A trade-off needs to be made.
  - Run Steps a,b,c on a commit.
  - Run Steps d & e once every 3 hours.
- How long does a Continuous Integration build run for?
  - One option to reduce time taken and ensure we have immediate feedback is to split the long running tests into a separate build which runs less often.

- **What is Continuous Delivery ?**
- Is it practice of delivering the software for testing as soon as it is build by CI (Continuous Integration) server's. It requires heavy use of Versioning Control System for so always available to developers and testers alike.
- As soon as the build is generated by the Continuous Integration (CI) servers, it will be delivered for testing. The practice of delivering the software for testing is known as Continuous Delivery.

- **What are the success factors for Continuous Integration?**
- Here you have to mention the requirements for Continuous Integration. You could include the following points in your answer:
- Maintain a code repository
- Automate the build
- Make the build self-testing
- Everyone commits to the baseline every day
- Every commit (to baseline) should be built
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment



Jenkins Interview Questions	
Jenkins is an	Open source software
Jenkins is an	Automation server
Jenkins can	Help to automate the software development process.
Jenkins can	Automate the process with continuous integration and facilitate technical aspects of continuous delivery.
Jenkins developed by	Jenkins is a fork of a project called Hudson.
Jenkins License	MIT
Jenkins has written in	Java

- **What is Jenkins?**
- Answer # Jenkins is an open source automation server. Jenkins is a continuous integration tool developed in Java. **Jenkins** helps to automate the non-human part of software development process, with **continuous integration** and facilitating technical aspects of continuous delivery.

- **Why do we use Jenkins?**
- Answer # Jenkins is an **open-source** continuous integration software tool written in the Java programming language for testing and reporting on isolated changes in a larger code base in real time. The **Jenkins software** enables developers to find and solve defects in a code base rapidly and to automate testing of their builds.

- **Another Answer:** Jenkins is mainly used to automate the code/test/build/deploy processes. The source code repository are integrated with jenkins to pick the code and test cases are run on Jenkins to test if any bug in code.

- If test cases are passed, codes are build on Jenkins using build plugins available in Jenkins itself. Finally, artifacts are created when build is successful. These artifacts are sent to Archive Repository Management (ARM) system or depolyed directly on target servers.

- Notification are being sent for any failure to respective developers. It smoothes the feedback system and error detection on early stage.

- **What is Maven and what is Jenkins?**
- Answer # **Maven is a build tool**, in short a successor of ant. It helps in build and version control. However, **Jenkins is continuous integration system**, where in maven is used for build. Jenkins can be used to automate the deployment process.

- **What is the difference between Hudson and Jenkins?**
- Answer # **Jenkins is the new Hudson**. It really is more like a rename, not a fork, since the whole development community moved to Jenkins. (Oracle is left sitting in a corner holding their old ball “**Hudson**“, but it’s just a soul-less project now.). In a nutshell **Jenkins CI** is the leading open-source continuous integration server.

- **What is the programming language used to build Jenkins?**
- A) Jenkins is an open source automation server written in Java.

- **What are the features of Jenkins?**
- A) Jenkins has many features like:
- Continuous Integration and Continuous Delivery - As an extensible automation server, Jenkins can be used as a simple CI server or turned into the continuous delivery hub for any project.
- Easy installation - Jenkins is a self-contained Java-based program, ready to run out-of-the-box, with packages for Windows, Mac OS X and other Unix-like operating systems.
- Easy configuration - Jenkins can be easily set up and configured via its web interface, which includes on-the-fly error checks and built-in help.
- Plugins - With hundreds of plugins in the Update Center, Jenkins integrates with practically every tool in the continuous integration and continuous delivery toolchain.
- Extensible - Jenkins can be extended via its plugin architecture, providing nearly infinite possibilities for what Jenkins can do.
- Distributed - Jenkins can easily distribute work across multiple machines, helping drive builds, tests and deployments across multiple platforms faster.

- **By using Jenkins, what are the benefits that one can have according to you?**
  The answer provided by you here depends largely upon the experience. If you don’t have much information regarding the Jenkins, you can reply the following.
- It is possible for the users to catch the build failure through Jenkins. Also, with this technology, it is possible to get notifications upon any single change made in the repository. Developers can simply get idea regarding the build report success or the failure and this can be done by the Jenkins integration with LDAP. In addition to this, the continuous integration agile development can also be assured simply. The best thing is Jenkins can simply let the users to look for the errors and major bugs in the early phase of development which later create a lot of issues.

- **What are the advantages of Jenkins?**
- Advantage of Jenkins includes:
- Bugs tracking are easy at early stage in development environment.
- Provides a large numbers of plugin support.
- Iterative improvement to the code.
- Build failures are cached at integration stage.
- For each code commit changes an automatic build report notification generates.
- To notify developers about build report success or failure, it is integrated with LDAP mail server.
- Achieves continuous integration agile development and test driven development.
- With simple steps, maven release project is automated.

- **Why do we use Jenkins with selenium?**
- Answer # Running **Selenium tests in Jenkins** allows you to run your tests every time your software changes and deploy the software to a new environment when the tests pass. Jenkins can schedule your tests to run at specific time.

- **What are CI Tools?**
- Answer # Here is the list of the top 8 **Continuous Integration tools**:
- Jenkins
- TeamCity
- Travis CI
- Go CD
- Bamboo
- GitLab CI
- CircleCI
- Codeship

- **What is a CI CD pipeline?**
- Answer # A **continuous integration** and deployment pipeline (**CD/CI**) is such an important aspect of a software project. It saves a ton of manual, error-prone deployment work. It results in higher quality software for continuous integration, **automated tests**, and code metrics.

- **What does CI CD stand for and for what purpose it is generally deployed in the software development environment?**
  It is basically an approach that stands for Continuous Integration and Deployment Pipeline. Generally, it is one of the widely used aspects of a software development approach and is useful when it comes to saving time and development that is free from all sort of errors. Simply the quality of the outcome can be enhanced with this approach and the good thing is there is several automation tools present in it with the help of which the users can easily keep up the pace without worrying about anything.

- **What is build** pipeline **in Jenkins?**
- Answer # Job chaining in **Jenkins** is the process of automatically starting other job(s) after the execution of a job. This approach lets you build **multi-step build pipelines** or trigger the rebuild of a project if one of its dependencies is updated.

- **What is a Jenkins pipeline?**
- Answer # The **Jenkins Pipeline plugin** is a game changer for Jenkins users. Based on a *Domain Specific Language (DSL)* in Groovy, the Pipeline plugin makes pipelines scriptable and it is an incredibly powerful way to develop complex, multi-step **DevOps pipelines**.
- Another Answer:
- Jenkins Pipeline (or simply "Pipeline") is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.

- **Can you write a simple Jenkins Pipeline Code for Java?**
- A) Here is the simple Jenkins Pipeline Code for Java:
- Jenkinsfile (Declarative Pipeline)

  ```
  pipeline {
      agent { docker 'maven:3.3.3' }
      stages {
          stage('build') {
              steps {
                  sh 'mvn --version'
              }
          }
      }
  }
  ```
-

- **What is a Continuous delivery pipeline?**
- A) A continuous delivery pipeline is an automated expression of your process for getting software from version control right through to your users and customers.
-

- **What is Declarative Pipeline in Jenkins?**
- A) Declarative Pipeline is a relatively recent addition to Jenkins Pipeline [1] which presents a more simplified and opinionated syntax on top of the Pipeline sub-systems.
- All valid Declarative Pipelines must be enclosed within a pipeline block, for example:

-
  ```
  pipeline {
      /* insert Declarative Pipeline here */
  }
  ```

- **What is a DSL Jenkins?**
- Answer # The Jenkins “Job DSL / Plugin” is made up of two parts: The Domain Specific Language (DSL) itself that allows users to describe jobs using a Groovy-based language, and a Jenkins plugin which manages the scripts and the updating of the Jenkins jobs which are created and maintained as a result.
-

- **What is the tool used for provisioning and configuration?**
- Answer # Ansible is an agent-less configuration management as well as orchestration tool. In Ansible, the configuration modules are called “Playbooks”. Like other tools, Ansible can be used for cloud provisioning.

- **What is the difference between Maven, Ant** and **Jenkins?**
- Answer # Maven and ANT are build tool but main difference is that maven also provides dependency management, standard project layout and project management. On difference between Maven, ANT and Jenkins, later is a continuous integration tool which is much more than build tool.

- **Which SCM tools Jenkins supports?**
- Answer # Jenkins supports version control tools, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, ClearCase and RTC, and can execute Apache Ant, Apache Maven and sbt based projects as well as arbitrary shell scripts and Windows batch commands.

- **How schedule a build in Jenkins?**
- Answer # In Jenkins, under the job configuration we can define various build triggers. Simple find the ‘Build Triggers’ section, and check the ‘ Build Periodically’ checkbox. With the periodically build you can schedule the build definition by the date or day of the week and the time to execute the build.
- The format of the ‘Schedule’ textbox is as follows:
- MINUTE (0-59), HOUR (0-23), DAY (1-31), MONTH (1-12), DAY OF THE WEEK (0-7)

- **Why do we use Pipelines in Jenkins?**
- Answer # Pipeline adds a powerful set of automation tools onto Jenkins, supporting use cases that span from simple continuous integration to comprehensive continuous delivery pipelines. By modeling a series of related tasks, users can take advantage of the many features of Pipeline:
- Code: Pipelines are implemented in code and typically checked into source control, giving teams the ability to edit, review, and iterate upon their delivery pipeline.
- Durable: Pipelines can survive both planned and unplanned restarts of the Jenkins master.
- Pausable: Pipelines can optionally stop and wait for human input or approval before continuing the Pipeline run.
- Versatile: Pipelines support complex real-world continuous delivery requirements, including the ability to fork/join, loop, and perform work in parallel.
- Extensible: The Pipeline plugin supports custom extensions to its DSL and multiple options for integration with other plugins.

- **What is a Jenkinsfile?**
- Answer # A Jenkinsfile is a text file that contains the definition of a Jenkins Pipeline and is checked into source control.
- Creating a Jenkinsfile, which is checked into source control, provides a number of immediate benefits:
- Code review/iteration on the Pipeline
- Audit trail for the Pipeline
- Single source of truth for the Pipeline, which can be viewed and edited by multiple members of the project.

- **How do you create Multibranch Pipeline in Jenkins?**
- Answer # The Multibranch Pipeline project type enables you to implement different Jenkinsfiles for different branches of the same project. In a Multibranch Pipeline project, Jenkins automatically discovers, manages and executes Pipelines for branches which contain a Jenkinsfile in source control.

- **What is blue ocean in Jenkins?**
- Answer # Blue Ocean is a project that rethinks the user experience of Jenkins, modelling and presenting the process of software delivery by surfacing information that’s important to development teams with as few clicks as possible, while still staying true to the extensibility that is core to Jenkins.
- Blue Ocean reevaluates the user experience of Jenkins. Outlined starting from the earliest stage for Jenkins Pipeline, yet at the same time perfect with freestyle jobs, Blue Ocean lessens clutter and builds clearness for each individual from the team. Blue Ocean's principle highlights include:
- Sophisticated visualization
- Pipeline editor
- Personalization
- Pinpoint precision
- Native integration for branch and pull requests.

- **What are the important plugins in Jenkins?**
- Answers # Here is the list of some important **Plugins in Jenkins**:
- Maven 2 project
- Git
- Amazon EC2
- HTML publisher
- Copy artifact
- Join
- Green Balls

- **What are Jobs in Jenkins?**
- Answer # **Jenkins** can be used to perform the typical build server work, such as doing continuous/official/nightly builds, run tests, or perform some repetitive batch tasks. This is called “**free-style software project**” in Jenkins.

- **How do you create a Job in Jenkins?**
- Answer # Go to **Jenkins** top page, select “New Job”, then choose “Build a free-style software project”. This job type consists of the following elements:
- optional **SCM**, such as **CVS** or **Subversion** where your source code resides.
  optional triggers to control when Jenkins will perform builds.
- some sort of build script that performs the build **(ant, maven, shell script, batch file**, etc.) where the real work happens optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
- optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc.

- **How do you configuring automatic builds in Jenkins?**
- Answer # **Builds in Jenkins** can be triggered periodically (on a schedule, specified in configuration), or when source changes in the project have been detected, or they can be automatically triggered by requesting the URL:
- http://YOURHOST/jenkins/job/PROJECTNAME/build

- **How to create a backup and copy files in Jenkins?**
- Answer # To create a backup, all you need to do is to periodically back up your **JENKINS_HOME** directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory.

- **What is the trustAnchors parameter must be non-empty error and how can you solve it?**
- A) This trustAnchors parameter must be non-empty error means that the truststore you specified was not found, or couldn’t be opened due to access permissions for example.
- EJP basically answered the question (and I realize this has an accepted answer) but I just dealt with this edge-case gotcha and wanted to immortalize my solution. I had the InvalidAlgorithmParameterException error on a hosted jira server that I had previously set up for SSL-only access.
- The issue was that I had set up my keystore in the PKCS#12 format, but my truststore was in the JKS format. In my case, I had edited my server.xml file to specify the keystoreType to PKCS, but did not specify the truststoreType, so it defaults to whatever the keystoreType is. Specifying the truststoreType explicitly as JKS solved it for me.

- **What are the feature differences between Jenkins and Hudson?**
- A) Jenkins is the recent fork by the core developers of Hudson. To understand why, you need to know the history of the project. It was originally open source and supported by Sun. Like much of what Sun did, it was fairly open, but there was a bit of benign neglect. The source, trackers, website, etc. were hosted by Sun on their relatively closed java.net platform.
- Then Oracle bought Sun. For various reasons Oracle has not been shy about leveraging what it perceives as its assets. Those include some control over the logistic platform of Hudson, and particularly control over the Hudson name. Many users and contributors weren’t comfortable with that and decided to leave.
- So it comes down to what Hudson vs Jenkins offers. Both Oracle’s Hudson and Jenkins have the code. Hudson has Oracle and Sonatype’s corporate support and the brand. Jenkins has most of the core developers, the community, and (so far) much more actual work.
- In fact, arguably it was Oracle who did the forking! And technically, too, that’s kinda what happened.
- It’s interesting to see what comes out of “Hudson” though. While the “Winston summarizes the state and rosy future of the Hudson project” stuff they posted on the (new) Hudson website originally seemed like odd humour to me, perhaps this was a purposeful takeover, and the Sonatype guys actually have some big ideas up their sleeve. This analysis, suggesting a deliberate strategy by Oracle/Sonatype to oust Kohsuke and crew to create a more “enterprisy” Hudson is a very interesting read!
- In any case, this brief comparison a fortnight after the split—while not exactly scientific—shows Jenkins to be by far more active of the two projects.

- Jenkins has continued the path well-trodden by the original Hudson with frequent releases including many minor updates.
- Oracle seems to have largely delegated work on the future path for Hudson to the Sonatype team, who has performed some significant changes, especially with respect to Maven. They have jointly moved it to the Eclipse foundation.
- I would suggest that if you like the sound of:
- Less frequent releases but ones that are more heavily tested for backwards compatibility (more of an “enterprise-style” release cycle)
- A product focused primarily on strong Maven and/or Nexus integration (i.e., you have no interest in Gradle and Artifactory etc)
- Professional support offerings from Sonatype or maybe Oracle in preference to Cloudbees etc
- You don’t mind having a smaller community of plugin developers etc.
  , then I would suggest Hudson.
- Conversely, if you prefer:
- More frequent updates, even if they require a bit more frequent tweaking and are perhaps slightly riskier in terms of compatibility (more of a “latest and greatest” release cycle)
- A system with more active community support for e.g., other build systems / artifact repositories
- Support offerings from the original creator et al. and/or you have no interest in professional support (e.g., you’re happy as long as you can get a fix in next week’s “latest and greatest”)
- A classical OSS-style witches’ brew of a development ecosystem
- then I would suggest Jenkins.

- **How to trigger a build remotely from Jenkins? How to configure Git post commit hook?**
- The requirement is whenever changes are made in the Git repository for a particular project it will automatically start Jenkins build for that project.
- A) As mentioned in “Polling must die: triggering Jenkins builds from a git hook”, you can notify Jenkins of a new commit:
- With the latest Git plugin 1.1.14 (that I just release now), you can now do this more >easily by simply executing the following command:
- curl http://yourserver/jenkins/git/notifyCommit?url=<URL of the Git repository>
  This will scan all the jobs that’s configured to check out the specified URL, and if they are also configured with polling, it’ll immediately trigger the polling (and if that finds a change worth a build, a build will be triggered in turn.)
- This allows a script to remain the same when jobs come and go in Jenkins.
  Or if you have multiple repositories under a single repository host application (such as Gitosis), you can share a single post-receive hook script with all the repositories. Finally, this URL doesn’t require authentication even for secured Jenkins, because the server doesn’t directly use anything that the client is sending. It runs polling to verify that there is a change, before it actually starts a build.
- As mentioned here, make sure to use the right address for your Jenkins server:
- since we’re running Jenkins as standalone Webserver on port 8080 the URL should have been without the /jenkins, like this:
- http://jenkins:8080/git/notifyCommit?url=git@gitserver:tools/common.git
  To reinforce that last point, ptha adds in the comments:

- It may be obvious, but I had issues with:
- curl http://yourserver/jenkins/git/notifyCommit?url=<URL of the Git repository>.
  The url parameter should match exactly what you have in Repository URL of your Jenkins job.
  When copying examples I left out the protocol, in our case ssh://, and it didn’t work.
- You can also use a simple post-receive hook like in “Push based builds using Jenkins and GIT”
- #!/bin/bash
  /usr/bin/curl –user USERNAME:PASS -s \
- http://jenkinsci/job/PROJECTNAME/build?token=1qaz2wsx
  Configure your Jenkins job to be able to “Trigger builds remotely” and use an authentication token (1qaz2wsx in this example).
- However, this is a project-specific script, and the author mentions a way to generalize it.
  The first solution is easier as it doesn’t depend on authentication or a specific project.

- I want to check in change set whether at least one java file is there the build should start.
  Suppose the developers changed only XML files or property files, then the build should not start.
- Basically, your build script can:
- put a ‘build’ notes (see git notes) on the first call on the subsequent calls, grab the list of commits between HEAD of your branch candidate for build and the commit referenced by the git notes ‘build’ (git show refs/notes/build): git diff –name-only SHA_build HEAD.
  your script can parse that list and decide if it needs to go on with the build.
  in any case, create/move your git notes ‘build’ to HEAD.

- **What is the agent directive in Jenkins?**
- A) The agent directive tells Jenkins where and how to execute the Pipeline, or subset thereof. As you might expect, the agent is required for all Pipelines.
- Underneath the hood, there are a few things agent causes to happen:
- All the steps contained within the block are queued for execution by Jenkins. As soon as an executor is available, the steps will begin to execute.
- A workspace is allocated which will contain files checked out from source control as well as any additional working files for the Pipeline.

- **What is agent in Jenkins?**
- A) The agent section specifies where the entire Pipeline, or a specific stage, will execute in the Jenkins environment depending on where the agent section is placed. The section must be defined at the top-level inside the pipeline block, but stage-level usage is optional.

- **What are Parameters in Jenkins?**
- A) In order to support the wide variety of use-cases Pipeline authors may have, the agent section supports a few different types of parameters. These parameters can be applied at the top-level of the pipeline block, or within each stage directive.
-

- **What is post?**
- A) The post section defines one or more additional steps that are run upon the completion of a Pipeline’s or stage’s run (depending on the location of the post section within the Pipeline).
  post can support one of the following post-condition blocks: always, changed, failure, success, unstable, and aborted. These condition blocks allow the execution of steps within the post section depending on the completion status of the Pipeline or stage.

- **What are stages?**
- A) Containing a sequence of one or more stage directives, the stages section is where the bulk of the "work" described by a Pipeline will be located. At a minimum it is recommended that stages contain at least one stage directive for each discrete part of the continuous delivery process, such as Build, Test, and Deploy.

- **What is environment directive?**
- A) The environment directive specifies a sequence of key-value pairs which will be defined as environment variables for the all steps, or stage-specific steps, depending on where the environment directive is located within the Pipeline.

- **What are triggers?**
- A) The triggers directive defines the automated ways in which the Pipeline should be re-triggered. For Pipelines which are integrated with a source such as GitHub or BitBucket, triggers may not be necessary as webhooks-based integration will likely already be present. The triggers currently available are cron, pollSCM and upstream.

- **What is input directive?**
- A) The input directive on a stage allows you to prompt for input, using the input step. The stage will pause after any options have been applied, and before entering the stage`s `agent or evaluating its when condition. If the input is approved, the stage will then continue. Any parameters provided as part of the input submission will be available in the environment for the rest of the stage.

- **What is Parallel in Jenkins?**
- A) Stages in Declarative Pipeline may declare a number of nested stages within them, which will be executed in parallel. Note that a stage must have one and only one of either steps or parallel. The nested stages cannot contain further parallel stages themselves, but otherwise behave the same as any other stage. Any stage containing parallel cannot contain agent or tools, since those are not relevant without steps.

- **What is Scripted Pipeline in Jenkins?**
- A) Scripted Pipeline, like Declarative Pipeline, is built on top of the underlying Pipeline sub-system. Unlike Declarative, Scripted Pipeline is effectively a general purpose DSL [2] built with Groovy. Most functionality provided by the Groovy language is made available to users of Scripted Pipeline, which means it can be a very expressive and flexible tool with which one can author continuous delivery pipelines.
-

- **What is Flow Control in Jenkins?**
- A) Scripted Pipeline is serially executed from the top of a Jenkinsfile downwards, like most traditional scripts in Groovy or other languages.

- **How to make sure that your project builds doesn?t break in Jenkins?**
- You must follow these steps to make sure that your project builds doesn?t break in Jenkins:
- First, perform successful clean install on your local machine with all unit tests.
- Check all your code changes.
- Synchronize with repository to make sure that all required config and POM changes and any difference is checked into the repository.

- **How can you move or copy Jenkins from one server to another?**
- Follow these steps to move or copy Jenkins from one server to another:
- First, copy the related job directory and slide a job from one installation of Jenkins to another.
- Make a copy of an already existing job by making clone of a job directory by a different name.
- Renaming an existing job by rename a directory.

- **Which commands can be used to start Jenkins manually?**
- You can use any one of the following commands to start Jenkins manually:
- (Jenkins_url)/restart: Forces a restart without waiting for builds to complete.
- (Jenkin_url)/safeRestart: Allows all running builds to complete.

- **What admin work we need to do on Jenkins?**
- There are many works for admin to do on Jenkins. Some of them are:
  1. Configure Jenkins
  2. Create/manage Users
  3. Install/Update plugins
  4. Create jobs/projects
  5. Taking back up of Jenkins
  6. Jenkins slave configuration
  7. Fix the failure jobs

- **What is SonarQube**
  A:Sonar is a web based code quality analysis tool for Maven based Java projects. It covers a wide area of code quality check points which include: Architecture & Design, Complexity, Duplications, Coding Rules, Potential Bugs, Unit Test etc.

How to integrate sonarcube with jenkins

- **SonarQube** is a web-based application which is used for centralized management of code quality. We decided to integrate it with Jenkins to provide a one click solution.
- **Scenario**: Integrate SonarQube with Jenkins to run unit test cases and publish results to SonarQube.
- Here is the **step-by-step procedure** to perform the scenario:
- Setup a Jenkins server if already not using.
- Goto plugin-manager of Jenkins to install “SonarQube Plugin”.
- Goto “System-configuration” of Jenkins to provide “SonarQube” server’s details .

- Create a Jenkins job and choose one source code management option (say git).
- Under build, add “Execute Shell” as build step and write commands to run unit tests. For example:
- #!/bin/bash
- cd /path/to/code/
- grails clean-all --non-interactive --plain-output;
- grails refresh-dependencies;
- grails -Dgrails.env=test test-app:unit --non-interactive --plain-output
- grails test-app -coverage -xml
-
-

- Add “Invoke Standalone SonarQube Analysis” as another build step and add below lines to “Analysis properties” block:

  sonar.projectKey=App Name- Any Identifier
- sonar.projectName=Project1
- sonar.projectVersion=1.0.0
- sonar.projectDescription=Static analysis for the AppName
- sonar.sources=path/to/code/src, path/to/code/grails-app
- sonar.groovy.cobertura.reportPath=path/to/code/target/test-reports/cobertura/coverage.xml
- sonar.language=grvy
- sonar.sourceEncoding=UTF-8
- Now, run the Grails application job and results could be seen on SonarQube server after job completion.

- **As a user, what are the requirements you need to fulfill for using the Jenkins Technology?**
  All the users must have a working build script that should be present in the repository and in addition to this, a source code repository accessible to the user should also be there. There are no other requirements.

- **Is it possible for the users to move the Jenkins from one server to another without changing its content?**

  Yes, it is possible provided the servers run on the same OS. In case they are different, this task can still be performed but it needs more care in terms of data handling.

- **How well do you know about Jenkins and how do you think it is different from the other tools in its class?**
  It is basically one of the popular tools for Software development automation. One of the best things about this tool is it has inbuilt plug-in for continuous integration. It is known to keep a close eye on the version control system, to simply let the users make sure of monitoring the build systems. Generally, a lot of changes are there which it notes down for the future references. The entire process is noted and the users are free to check whether the changes made are appropriate or not.

- **Name any two important Plugging that you can deploy in Jenkins while handling software development automation?**

  These are Amazon EC2, Copy Artifact and HTML Publisher

- **Tell what should you do for moving the Jenkins between different servers?**
  First of all copy the related job directory from the original set of installation to another. Then, users need to make a copy of a job present already simply by making use of the cloning option but with a different name. The job remaining can now be copied directly to another server.

- **Can you start Jenkins manually? Tell the commands you use for this if so?**
  Yes, it is possible and for this there are two commands present and they are Jenkin_url/restart and Jenkin _url/safe restart

- **What are the conditions to deploy a custom build for a core plug-in?**
  First of all, Jenkins are to be stopped, next condition is to delete the history of plug-in expanded recently. An empty file with a default name is to be created post which the users have to start Jenkins.

- **How one can create a backup in the Jenkins?**
  One of the best things about Jenkins is it simply store all the data including the Build artifacts, as well as logs in it. In case a back of the setup and all other information is required, the users have to make sure that they have enabled the cloning of job directory or replicate the same post which backup can be created with default options.

- **Tell us a bit about what your service and plugin do. Who is it for? What are the highlights of your plugin?**
  LOADER.IO is a simple-to-use cloud-based load testing service. The service is designed for developers and people who need to ensure applications are performing as they should. It allows developers to perform large-scale load tests on demand, which lets them understand the scalability and performance of their applications. We realize Jenkins is the preferred build service for a lot of our users, and we know providing a way for them to implement, measure and improve application performance during the continuous build cycle is important. So we wrote a Jenkins plugin that allows load testing to be brought into the continuous build and deployment process with ease.

- **Explain how you can move or copy Jenkins from one server to another?**
- I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that; I have mentioned them below:
  You can:
- Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
- Make a copy of an existing job by making a clone of a job directory by a different name.
- Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

- **Explain how can create a backup and copy files in Jenkins?**
- Answer to this question is really direct. To create a backup, all you need to do is to periodically back up your JENKINS_HOME directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

- **Explain how you can setup Jenkins job?**
- My approach to this answer will be to first mention how to create Jenkins job. Go to Jenkins top page, select “New Job”, then choose “Build a free-style software project”.
  Then you can tell the elements of this freestyle job:
- Optional SCM, such as CVS or Subversion where your source code resides.
- Optional triggers to control when Jenkins will perform builds.
- Some sort of build script that performs the build (ant, maven, shell script, batch file, etc.) where the real work happens.
- Optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
- Optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc..

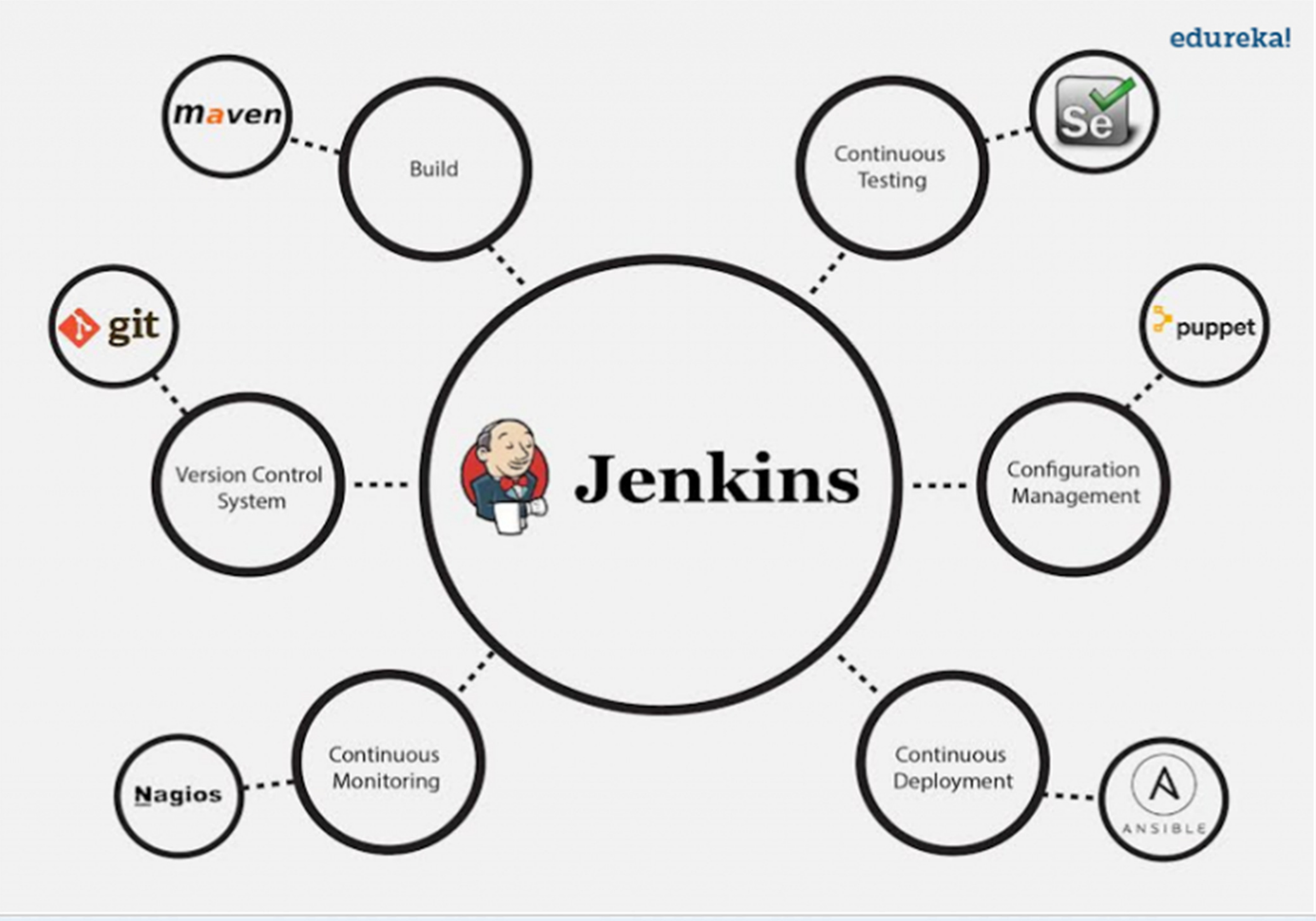
- **Mention some of the useful plugins in Jenkins.**
- Below, I have mentioned some important Plugins:
- Maven 2 project
- Amazon EC2
- HTML publisher
- Copy artifact
- Join
- Green Balls
- These Plugins, I feel are the most useful plugins. If you want to include any other Plugin that is not mentioned above, you can add them as well. But, make sure you first mention the above stated plugins and then add your own.

- **How will you secure Jenkins?**
- The way I secure Jenkins is mentioned below. If you have any other way of doing it, please mention it in the comments section below:
- Ensure global security is on.
- Ensure that Jenkins is integrated with my company’s user directory with appropriate plugin.
- Ensure that matrix/Project matrix is enabled to fine tune access.
- Automate the process of setting rights/privileges in Jenkins with custom version controlled script.
- Limit physical access to Jenkins data/folders.
- Periodically run security audits on same.

- **What are the benefits of using Jenkins?**
- I will suggest you to include the following benefits of Jenkins, if you can recall any other benefit apart from the below mentioned points you can include that as well.
- At integration stage, build failures are cached.
- For each change in the source code an automatic build report notification is generated.
- To notify developers about build report success or failure, it is integrated with LDAP mail server.
- Achieves continuous integration agile development and test driven development.
- With simple steps, maven release project is automated.
- Easy tracking of bugs at early stage in development environment than production.

- **What are the pre-requisites for using Jenkins?**
- Answer to this is pretty straightforward To use Jenkins you require:
- A source code repository which is accessible, for instance, a Git repository.
- A working build script, e.g., a Maven script, checked into the repository.
- *Remember, you have mentioned Plugins in your previous answer, so next question in this Jenkins interview questions blog will be regarding Plugins.*

- **Mention some of the useful plugins in Jenkins?**
- Below I have mentioned some important Plugins:
- Maven 2 project
- Git
- Amazon EC2
- HTML publisher
- Copy artifact
- Join
- Green Balls
- These Plugins I feel are the most useful plugins, if you want to include any other Plugin that is not mentioned above, you can add that as well, but make sure you first mention the above stated plugins and then add your own.


edureka!
maven
Build
Continuous Testing
Se
git
Version Control System
Jenkins
puppet
Configuration Management
Nagios
Continuous Monitoring
Continuous Deployment
ANSIBLE

- **Mention what are the commands you can use to start Jenkins manually?**
- For this answer I will suggest you to go with the below mentioned flow:

- To start Jenkins manually open Console/Command line, then go to your Jenkins installation directory. Over there you can use the below commands:
- To start Jenkins: **jenkins.exe start**
  To stop Jenkins: **jenkins.exe stop**
  To restart Jenkins: **jenkins.exe restart**

- **Explain how you can deploy a custom build of a core plugin?**
- Below are the steps to deploy a custom build of a core plugin:
- Stop Jenkins.
- Copy the custom HPI to **$Jenkins_Home/plugins**.
- Delete the previously expanded plugin directory.
- Make an empty file called **<plugin>.hpi.pinned**.
- Start Jenkins.

- **What is the relation between Hudson and Jenkins?**
- You can just say Hudson was the earlier name and version of current Jenkins. After some issue, the project name was changed from Hudson to Jenkins.

- **What you do when you see a broken build for your project in Jenkins?**
- There can be multiple answers to this question I will approach this task in the following way:
- I will open the console output for the broken build and try to see if any file changes were missed. If I am unable to find the issue that way, then I will clean and update my local workspace to replicate the problem on my local and try to solve it.
- If you do it in a different way then just mention that in your answer.

- **Explain how you can move or copy Jenkins from one server to another?**
- I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that, I have mentioned it below:

- You can:
- Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
- Make a copy of an existing job by making a clone of a job directory by a different name.
- Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

- **What are the various ways in which build can be scheduled in Jenkins?**
- You can schedule a build in Jenkins in the following ways:
- By source code management commits
- After completion of other builds
- Can be scheduled to run at specified time (crons)
- Manual Build Requests

- **What is the difference between Maven, Ant and Jenkins?**
- Maven and Ant are Build Technologies whereas Jenkins is a continuous integration tool.

- **Which SCM tools Jenkins supports?**
- Below are Source code management tools supported by Jenkins:
- AccuRev
- CVS,
- Subversion,
- Git,
- Mercurial,
- Perforce,
- Clearcase
- RTC

- **What are the two components Jenkins is mainly integrated with?**
- According to me Jenkins is mainly integrated with the following:
- Version Control system like GIT,SVN.
- Build tools like Apache Maven.

- **How can you clone a Git repository via Jenkins?**
- If you want to clone a Git repository via Jenkins, you have to enter the e-mail and user name for your Jenkins system. Switch into your job directory and execute the "git config" command for that.

- **How can you setup Jenkins jobs?**
- Follow these steps:
- Select new item from the menu.
- After that enter a name for the job and select free-style job.
- Then click OK to create new job in Jenkins.
- The next page enables you to configure your job.

- **What are the two components Jenkins is mainly integrated with?**
- Jenkins is integrated with these two components:
- Version Control system like GIT,SVN
- And build tools like Apache Maven.

- **What is the minimum JRE required to run jenkins 2.1?**
- JDK 8 or JRE 8

- **What is the use of Jenkins in Devops?**
- Jenkins automate the CI / CD process in Devops layer.

- **What is CI and CD process?**
- Continuous integration
  Developers practicing continuous integration merge their changes back to the main branch as often as possible
- Continuous delivery
  Continuous delivery is an extension of continuous integration to make sure that you can release new changes to your customers quickly in a sustainable way.

- **How do you back up your jenkins job & master configs?**
  A: Check in the JENKINS_HOME dir into source control (with appropriate ignore rules for the workspace directories and artifacts which do not need to be in source control.

- **How do you fetch a job's artifacts from a remote system? Do you need a plugin to do this?**
  A: Use the jenkins workspace URL with the archive artifacts plugin.

- **What are the different job permissions possible in Jenkins, and what are the reasons you would use each?**

  A: I would look for an explanation of:

  -Open (no permissions)

  -Logged in - all are admins

  -Matrix based

  -Project based matrix

- **How do you connect a build node to the Jenkins master with out using a web browser or being in front of the physical machine?**

  A: Using the JNLP service and PSTools command line to install the service

- **How to setup / configure Jenkins jobs?**
  Steps to be followed for settingup Jenkins jobs:
- Go --> Jenkins Dashboard --> Click on New Item
- Select the ‘Freestyle project option’.
- Select OK button inorder to create a new job.
- Next Page Displays, where you have to configure the job.

- **Can you give some idea about the tools which are generally used for the configuration and provisioning in the Jenkins?**
  The tool which is generally preferred is Ansible and it has mainly features for the cloud deployment of the automated software development approach. This tool has many additional features with it and the best thing is it can easily be deployed as an orchestration tool.

- **Can you make sure of dependency management in the Maven and ANT?**
  In case of Maven, the same is possible whereas, it is not possible in case of Ant. This is also a basic difference between both of them

- **How can you schedule a Build in the Jenkins environment?**
  It is possible to do so either manually or with the Build triggers. Users simply need to look for the concerned Build Triggers and then need to make sure that task is repeated periodically. The build definition can easily be scheduled without facing any error if the time to execute the build is already defined.

- **Name Some Pipeline Development Tools?**
- The Blue Ocean Pipeline Editor
- Command-line Pipeline Linter
- Replay

- **Define Fingerprint in Jenkins?**
- A hash is a unique globally, that tracks usgae of an Artifact or some other entities across the mutiple pipelines.

- **What are the actual applications for which the Jenkins can be deployed?**
  It is basically an approach that is used when it comes to testing and reporting the isolated changes that generally take place in the codes which are complex and are real time. The developers can easily make sure that the defects can be avoided and the best part is the quality of the overall code can be improved rapidly and all the builds can be tested through an automated approach.

- **When it comes to Software development, do you think the testing can be automated or it should be performed manually?**
  Most of the testing procedures can simply be considered through automated technologies depending upon the scale or level upto which it is required. However, a few little procedures are still opted manually for proper bug elimination and control.

- **As a user, when you will use the Jenkins with the Selenium and what is the scope of the same?** Well, the fact is Selenium tests are quite common in the Jenkins environment and the users are free to keep up the pace simply with them whenever there is a change taken place in the software. With Jenkins, the test can be scheduled to run in a way that is required by the user. It is possible to run any test anytime and they can even be made to run at a later date.

- **What are the Continuous Integration tools you are familiar with?**
  Before you answer these questions, keep it in mind that generally the users are not familiar with all of them. In other words, even if you use Jenkins, it’s not necessary that you use all of them even when the tasks are quite complex. So you can tell the interviewer about the ones you are familiar with from the below list.
- Bamboo, Go CD, TeamCity, CodeShip, GitLab CI, Travis CI, Circle CI.

- **Compare Jenkins with Maven? And tell how you can put them separate from one another.** Maven is actually a build tool which is useful for build and control purposes while on the other side, the Jenkins is a powerful continuous integration approach that can automate the development processes in very reliable manner. Maven is preferred while the applications are short scale whereas when the code is large and needs a lot of attention, the Jenkins approach is adopted and preferred.

- **How it is possible for the users to access the multiple Jenkinfiles that belong to different branches and have the same project associated with them?**
  This is done generally with the help of Multibranch Pipeline concept where the Jenkin can automatically block for the pipelines and manage them.

- **What do you mean by Repository?**
  In the computer world, a repository is a central storage in which an accumulation of data and is kept and maintained in an organized way, usually in server storage. The term has been derived is from the Latin repositorium, which means a vessel or chamber in which things can be placed, where anyone can collect them. Depending on how the repository will be used, a repository may be directly accessible to users or may be a place from which specific databases, files, or documents which are pulled for further development, relocation or distribution. A repository may be assembling of data itself into some accessible place of storage or it may also imply some ability to selectively extract data.

- **What does 'Nightly Builds' mean?**
  It means an automated build that is routinely done once in a day, after the end of the day for most of the developers. For projects with developers in several time zones, it is usually a compromise time. The idea is that everyone who checks in code "today", the automated build ensures that everything compiles, and also runs the unit tests and other automated tests that has been programmed to produce a final installer/executable etc.

- **How can we configure Nightly Builds?**
  We can click on a new job and we should name the job in the source code control management, which we wish to give the subversion and then give the subversion URL where we want to deploy the subversion where we have the source code. Then we have the building triggers category which contains pole scm and build periodically. Pole scm means if we want to do a build for every check in then we have to use pole scm, if we want to use build periodically or run a build on a particular time and all that we have to use build periodically and we give crone job syntax which contains the build which has to execute the shell and invoke the ant and invoke maven, and if we want an invoke ant spirit we can select invoke ant and then pass the corresponding target to post bill actions which will contain email notifications and if the build fails then we will automatically get an email notifications for post build activities.

- **Difference between jar, war and ear?**
  Jar is Java Archive i.e. compressed Class or Class / Java files. War comprises of compressed Servlet class files, JSP FIles, supporting files, GIF and HTML files and Ear comprise of compressed Java and web module files (was files).

- **What is the connection between master and slave?**

- In a standard Jenkins master/slave setup, Jenkins is only installed on the master. That is where you see the user interface and start/configure build jobs.
- The slaves execute the jobs.
-

  There is no Jenkins installation here other than a small Java app to have Jenkins communicate to/from the slave. Jenkins talks to these slaves through the slave.jar app over e.g. SSH via the SSH Slaves Plugin and can monitor if the slave is running, etc.