CI/CD - Interview questions - PART 2

Created	@November 5, 2023 1:19 PM
Tags	CI/CD DevOps Interview Q&A
Topic	**CI/CD - Jenkins & Github Actions**
Youtube channel	DevOps Motivation

Jenkins

What is Jenkins? What have you used it for?

Jenkins is a Java-based open-source tool for continuous integration, helping developers build, test, and deliver software efficiently through various automation plugins.
Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis and much more.

What are the advantages of Jenkins over its competitors? Can you compare it to one of the following systems?

- Travis
- Bamboo
- Teamcity
- CircleCI

Jenkins has several advantages over its competitors, including Travis, Bamboo, TeamCity, and CircleCI. Here are some of the key advantages:

1. Open-source and free
2. Customizable and flexible
3. Wide range of integrations and Plugins
4. Active and supportive community

When comparing Jenkins to its competitors, there are some key differences in terms of features and capabilities. For example:

Travis: A user-friendly cloud-based CI/CD platform with quick setup, but limited customization and integrations compared to Jenkins.

Bamboo: Atlassian's CI/CD tool with extensive features, but it can be pricier and more complex to configure than Jenkins.

TeamCity: JetBrains' CI/CD tool offering robust features, but it can be complex and resource-intensive in comparison to Jenkins.

CircleCI: A cloud-based CI/CD platform with fast build times and GitHub integration, though it can be costlier, especially for larger projects, compared to Jenkins.

What are the limitations or disadvantages of Jenkins?

This might be considered to be an opinionated answer:

Old fashioned dashboards with not many options to customize it
Containers readiness (this has improved with Jenkins X)
By itself, it doesn't have many features.
On the other hand, there many plugins created by the community to expand its abilities

Managing Jenkins and its pipelines as a code can be one hell of a nightmare

Explain the following:
• Job
• Build
• Plugin

• Node or Worker
• Executor

Job is an automation definition = what and where to execute once the user clicks on "build"
Build is a running instance of a job. You can have one or more builds at any given point of time (unless limited by configuration)
A worker is the machine/instance on which the build is running. When a build starts, it "acquires" a worker out of a pool to run on it.

An executor is variable of the worker, defining how many builds can run on that worker in parallel. An executor value of 3 means, that 3 builds can run at any point on that executor (not necessarily of the same job. Any builds)

What plugins have you used in Jenkins?

Jenkins has a vast library of plugins, and the most commonly used plugins depend on the specific needs and requirements of each organization. However, here are some of the most popular and widely used plugins in Jenkins:

Pipeline: Enables users to create and manage multi-stage pipelines for automated software delivery.

Git: Integrates with Git for code version control, allowing pulling, triggering builds, and pushing code changes.

Docker: Integrates with Docker for building and running containers, streamlining application deployment.

JUnit: Integrates JUnit for running unit tests and generating test result reports.

Email Extension: Enhances email notifications with customization options and targeting specific recipients based on build results.

Artifactory: Integrates with Artifactory for storing and managing binary artifacts in the build process.

SonarQube: Integrates with SonarQube for code quality analysis, measuring code complexity, duplication, and coverage.

Have you used Jenkins for CI or CD processes? Can you describe them?

To set up a Jenkins-based CI/CD pipeline for a Node.js web app:

1. Install Jenkins: On a server or cloud platform like AWS.
2. Install necessary plugins (e.g., NodeJS, Git, Docker).
3. Create a new job to build the Node.js app.
4. Configure the job to pull Git code, install dependencies, run tests, and build.
5. Set up a deployment environment, e.g., with Docker.
6. Configure continuous deployment if tests pass.
7. Monitor and troubleshoot for any issues.

What type of jobs are there? Which types have you used?

In Jenkins, job types include:

1. Freestyle job: Customizable with build steps, triggers, SCM polling, and post-build actions.
2. Pipeline job: Defines a sequence of jobs using a script-like Jenkinsfile.
3. Multi-configuration job: Runs the job with multiple configurations, creating a matrix of results.
4. Maven job: Designed for Java applications, automates Maven build processes.
5. Parameterized job: Allows runtime customization by defining and passing parameters.

How did you report build results to users? What ways are there to report the results?

You can report via:
• Emails
• Messaging apps
• Dashboards

Each has its own disadvantages and advantages. Emails for example, if sent too often, can be eventually disregarded or ignored.

GitHub Actions

What is a Workflow in GitHub Actions?

A YAML file that defines the automation actions and instructions to execute upon a specific event.
The file is placed in the repository itself.

A Workflow can be anything - running tests, compiling code, building packages, ...

What is a Runner in GitHub Actions?

A workflow has to be executed somewhere. The environment where the workflow is executed is called Runner.A Runner can be an on-premise host or GitHub hosts

What is a Job in GitHub Actions?

A job is a series of steps which are executed on the same runner/environment.A workflow must include at least one job.

What is an Action in GitHub Actions?

An action is the smallest unit in a workflow. It includes the commands to execute as part of the job.

In GitHub Actions workflow, what the 'on' attribute/directive is used for?

Specify upon which events the workflow will be triggered.For example, you might configure the workflow to trigger every time a changed is pushed to the repository.

True or False? In Github Actions, jobs are executed in parallel by default

True

How to create dependencies between jobs so one job runs after another?

Using the "needs" attribute/directive.

```
jobs:
  job1:
  job2:
    needs: job1
```

In the above example, job1 must complete successfully before job2 runs

How to add a Workflow to a repository?

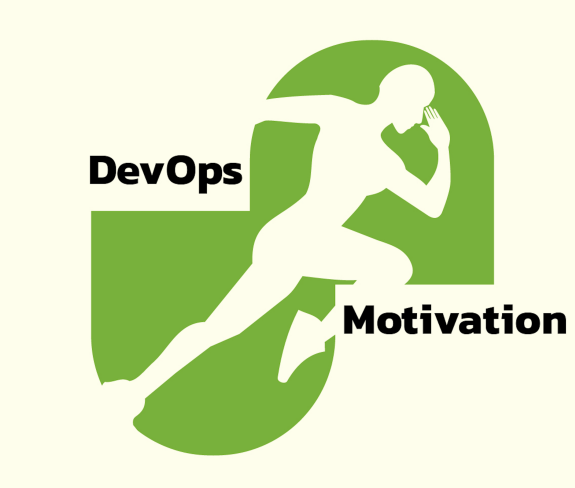
CLI:

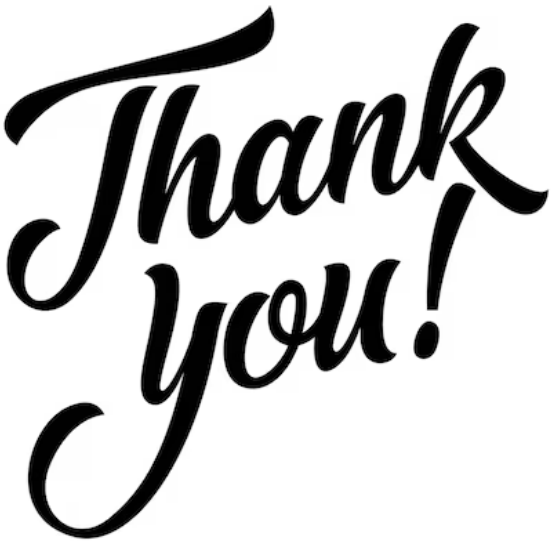
1. Create the directory `.github/workflows` in the repository

2. Add a YAML file

SUBSCRIBE TO OUR CHANNEL FOR MORE INTERESTING CONTENT !!

— Devops Motivation —




Thank you!