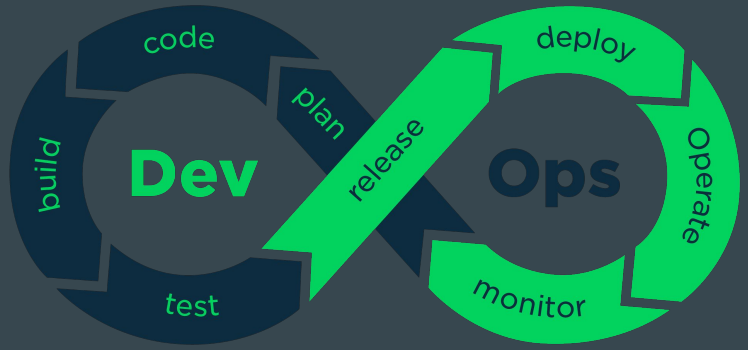
An Introduction to Jenkins

By Jacob Bonner

CI/CD

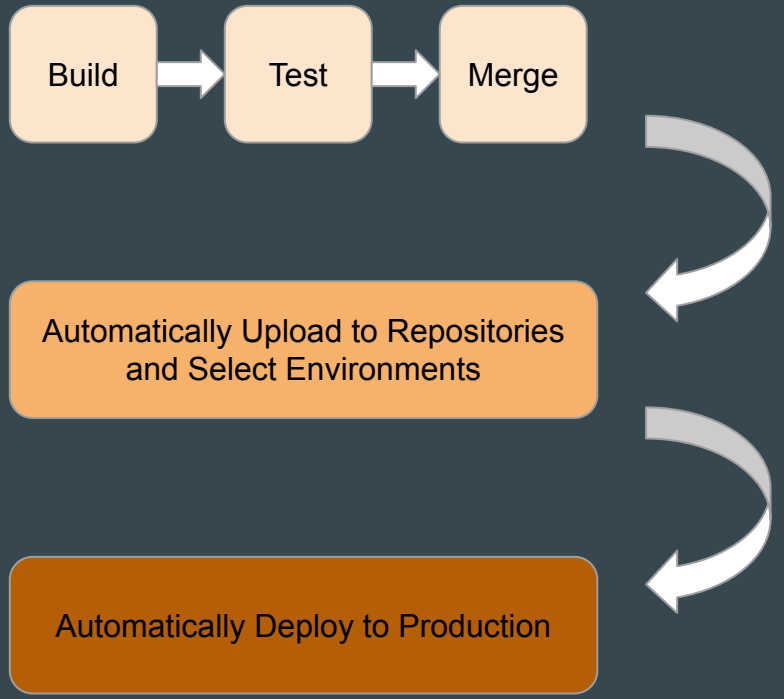
What is CI/CD?

- A method to frequently deliver applications by incorporating ongoing automation and monitoring during the lifecycle of an application
    - Includes stages of development from integration, functionality and performance testing, to delivery and deployment



CI/CD Pipeline

- Continuous Integration
  - Automating and integrating code changes from a team throughout development
- Continuous Delivery
  - Automatically uploading application changes to select environments or repositories
  - Testing of application in staging environment to ensure code integrity and deployability
- Continuous Deployment
  - Automatically release applications to production environments



Jenkins

What is Jenkins?

- Open source framework that facilitates CI/CD in software projects by automating various processes
  - Building, Testing, Deploying, etc

- Can be used as a simple CI server or extended into the CI/CD hub for any project

Why Use Jenkins? Free / Open Source

- Community
  - Large support and thorough documentation
  - Conferences and events
- Anyone can examine the code
  - Develop new features
  - Find and fix issues
- Plugins
  - Hundreds available to support building, deploying and automating any project
  - Easy to create your own

Why Use Jenkins? Extensibility and Flexibility

- Hundreds of community developed plugins
  - Allows integration with practically every tool in the CI/CD toolchain.
  - Extends features and functionality, providing limitless possibilities

- Distributed
  - Easily distribute work across multiple platforms and machines, driving builds, tests and deployments faster

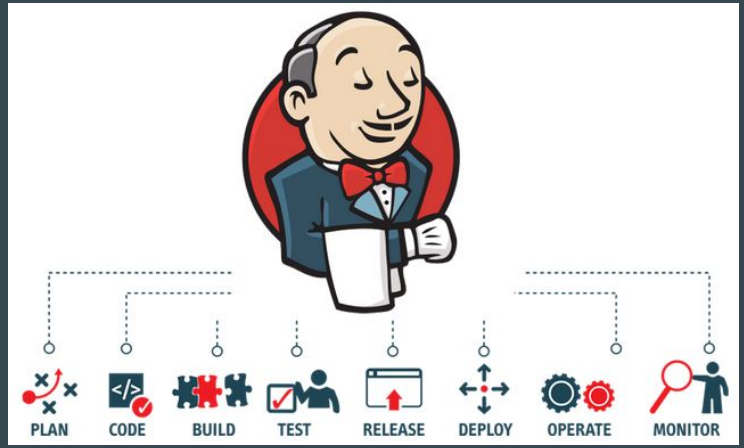
Why Use Jenkins? Ease of Use

- Easy Installation
  - Self-contained and java-based, ready to run out-of-the-box for all major operating systems
- Easy Configuration
  - Set up and configured via its web interface
    - Includes on-the-fly error checks and built-in help
  - Documentation and examples directly included or easily accessible
- User-friendly Interface
  - Intuitive
  - Easy to navigate

What can Jenkins do?

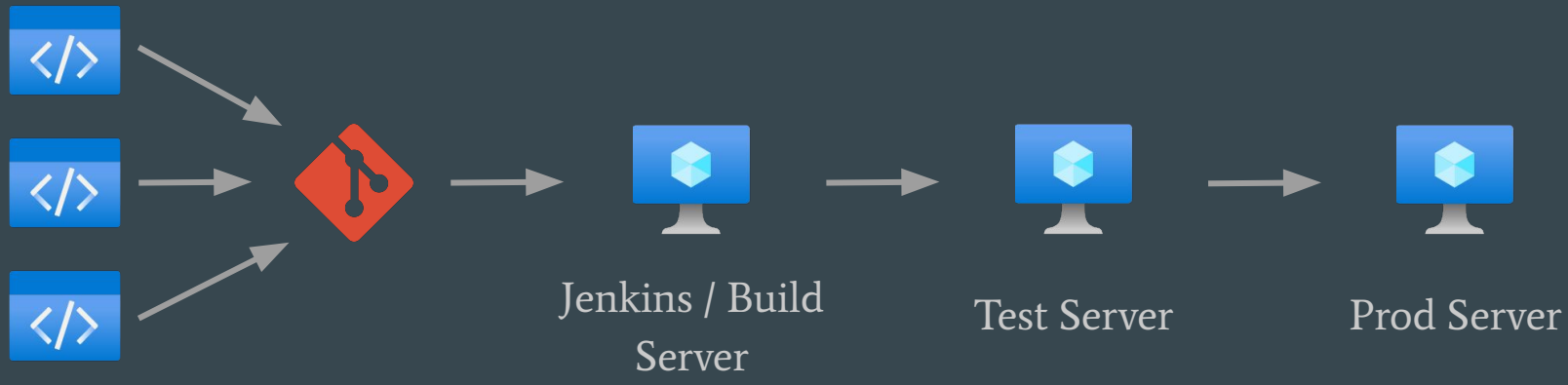
- Integrate with various SCM systems
- Maintain/Manage, orchestrate, and accelerate processes throughout the entire software development life cycle
    - Software builds using various build systems
    - Automation testing using different test frameworks
    - Execute Test Scripts
    - Achieve test results and perform post actions
    - Document, Package, Deploy, etc

And, with the help of plugins, so much more!



Sample Jenkins Architecture

1. Developers make modifications to source code and commit to a repository.
2. The Jenkins server pulls changes from the repository.
3. The Build server performs a build with the new code.
4. On a successful build, the application is deployed to a Test server.
5. If no testing issues, the application changes are deployed to the Production server.



Terminology

- Job/Project
  - A user-configured description of work which Jenkins should perform, such as building a piece of software, etc.
- Pipeline
  - A user-defined model of a continuous delivery pipeline.
- Item
  - An entity in the web UI corresponding to either a Folder, Pipeline, or Job/Project.
- Build
  - The result of a single execution of a Job/Project.

Terminology

- Controller
  - The central, coordinating process which stores configuration, loads plugins, and renders the various user interfaces for Jenkins.
- Agent
  - Typically a machine, or container, which connects to a Jenkins controller and executes tasks when directed by the controller.
- Node
  - A machine which is part of the Jenkins environment and capable of executing Pipelines or Projects. Both the Controller and Agents are considered to be Nodes.

Controller-Agent Architecture

Jenkins Controller

- Schedule builds and dispatch them to appropriate agents
- Monitor agents and take them online/offline
- Present build results, reports, and artifacts
- Can execute Jobs/Projects directly but it is best practice to select the appropriate agent(s) for build and execution-related tasks

Jenkins Agent(s)

- Listen to commands and execute builds as directed by the Controller
- Sends results, reports, or artifacts back to the Controller
- By default, the Controller selects the best-suited Agent for a Job/Project, but Jenkins users can specify an Agent or a particular type of Agent

Jenkins User Interface

- New Item
  - Create a new Item, such as a project/job.
- People
  - Configure and view a list of known users
- Build History
  - A list of the builds that have completed
- Manage Jenkins
  - System Configuration
  - Security

- Build Queue
  - A list of what builds are waiting to be run

- Build Executor
  - Shows what builds are running at the present moment

Freestyle Projects/Jobs

- What is a Project/Job?
  - A user-configured description of work which Jenkins should perform.
- Configuration
  - Adding parameters, disabling the job, etc
  - Controlling how Jenkins interacts with any code stored in external SCM
  - Specifying how and when a job is built automatically
  - Controlling the environment in which the job will run
  - Adding various types of steps that will be executed in any build of the job
  - Adding various steps that will execute after a build of the job is complete

Jenkins Pipeline

- What is Pipeline?
  - A user-defined model of a continuous delivery pipeline.
  - “Pipeline-as-code” written in a ‘Jenkinsfile’
- Why Pipeline?
  - Code, Durable, Pausable, Versatile, Extensible
- Concepts and Syntax
  - **Pipeline**: User-defined pipeline as a whole, representing the basis for your entire build process.
  - **Node/Agent**: A machine that is part of the Jenkins environment. Capable of building a Pipeline.
  - **Stage**: Defines a conceptually distinct subset of tasks performed through the entire Pipeline
  - **Step**: A single task, telling Jenkins what to do at a particular point in time / step in the process.

Sample Pipeline

```
pipeline {
    agent any // specifies the agent to run on, which in this case is any agent
    stages {
        stage('Build') { // Defines the Build stage
            steps {
                // perform any steps required for the Build stage
            }
        }
        stage('Test') { // Defines the Test stage
            steps {
                // perform any steps required for the Test stage
            }
        }
        stage('Deploy') { // Defines the Deploy stage
            steps {
                // perform any steps required for the Deploy stage
            }
        }
    }
}
```

Parameters, Global Variables, Workspaces

- Parameters
  - Boolean, String, Choice, Credentials, etc
- Global Variables
  - Environment Variables
  - Parameters
  - Properties of the current build of a Pipeline
  - SCM configuration
- Workspace
  - A disposable directory on the file system of a Node where work can be done by a Pipeline/Job
  - Every Job and Pipeline you define is given a dedicated workspace

Builds

- **Build**: The result of a single execution of a Job/Project.
- Tracking/Monitoring Build State
  - It is useful to be able to go into the system and monitor the current status of a build environment
- Build Artifacts
  - An artifact is an immutable file generated during a build which is archived onto the Controller
- Polling SCM for Build Triggering
  - It is important in CI/CD to communicate with a SCM system, like Github
- Triggering Builds with GitHub Webhooks
  - The goal is to integrate Jenkins with SCM, so that changes in a repository, are automatically pushed back into the Jenkins environment.

Further Topics in My Tutorial

- Agents and Distributing Builds
  - Adding an SSH Build Agent
  - Using Docker Image for Agents
- Testing and Post-Execution Behavior
- Jenkins REST API
  - Trigger builds and retrieve various information from Jenkins
- Further Topics
  - Using Folders and Views to organize jobs
  - Triggering Downstream Builds
  - Security
  - Finding and using more Plugins
  - And more!

Works Cited

"GitHub Jenkins Plugin." *GitHub*, https://plugins.jenkins.io/github/.

Inconshreveable. "Documentation." *Ngrok*, https://ngrok.com/docs#getting-started.

Jenkins, https://www.jenkins.io/.

"What Is CI/CD, Continuous Integration and Continuous Delivery?" *Cisco*, Cisco, 22 Mar. 2021, https://www.cisco.com/c/en/us/solutions/data-center/data-center-networking/what-is-ci-cd.html.

"What Is CI/CD?" *Red Hat*, 31 Jan. 2018, https://www.redhat.com/en/topics/devops/what-is-ci-cd.

"What Is the CI/CD Pipeline?" *IBM*, 27 Sep. 2021, https://www.ibm.com/cloud/blog/ci-cd-pipeline.