For beginners…

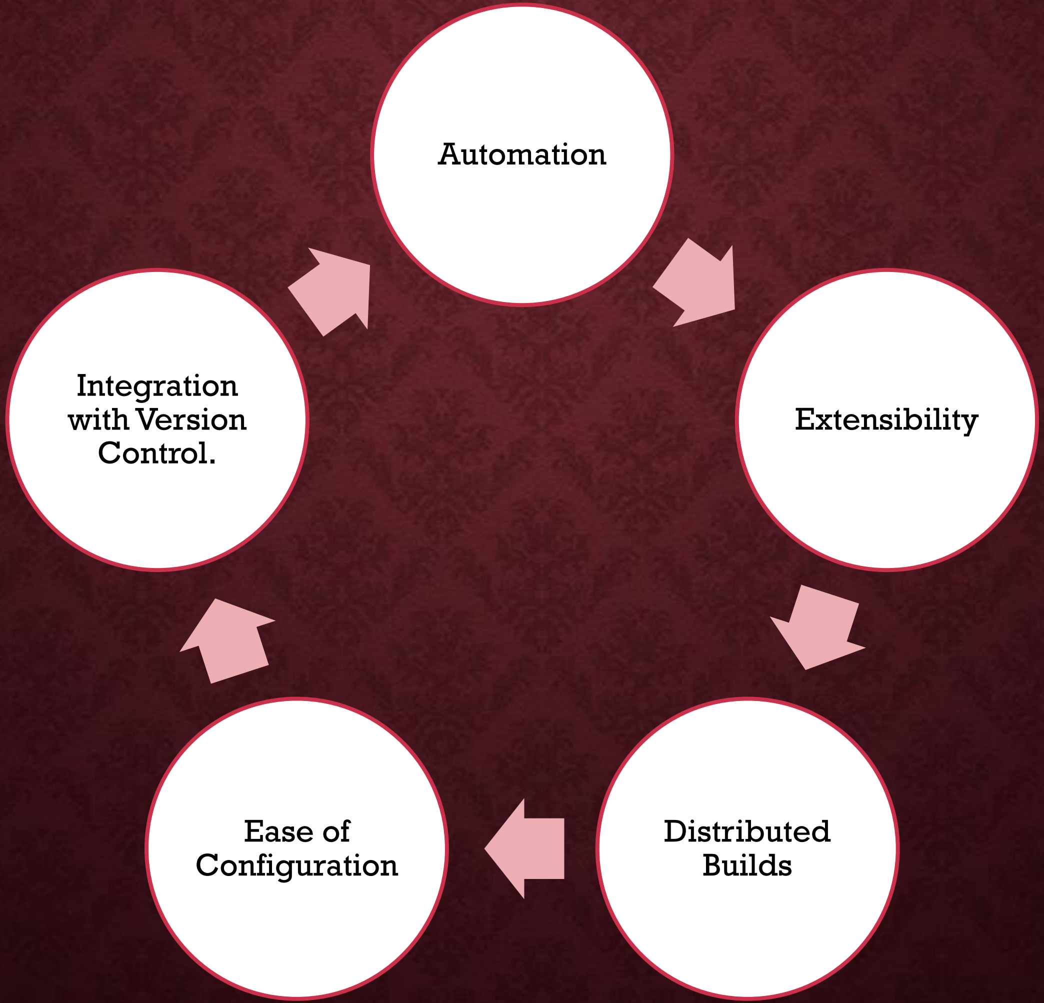
INTRODUCTION TO JENKINS

- Jenkins is an open-source automation server widely used for building, testing, and deploying software.
- It facilitates continuous integration and continuous delivery (CI/CD) by automating the building, testing, and deployment processes of software development.
- It is a server-based system that runs in servlet containers such as Apache Tomcat.

MORE ABOUT JENKINS…

- The primary purpose of Jenkins is to streamline the software development lifecycle by automating repetitive tasks involved in the integration and delivery of code changes.
- It helps teams deliver high-quality software faster and more reliably.
- Jenkins has a vibrant and active community, with a wealth of documentation, forums, and user groups.
- Users can find support, share experiences, and contribute to the Jenkins community.

KEY FEATURES OF JENKINS



Automation

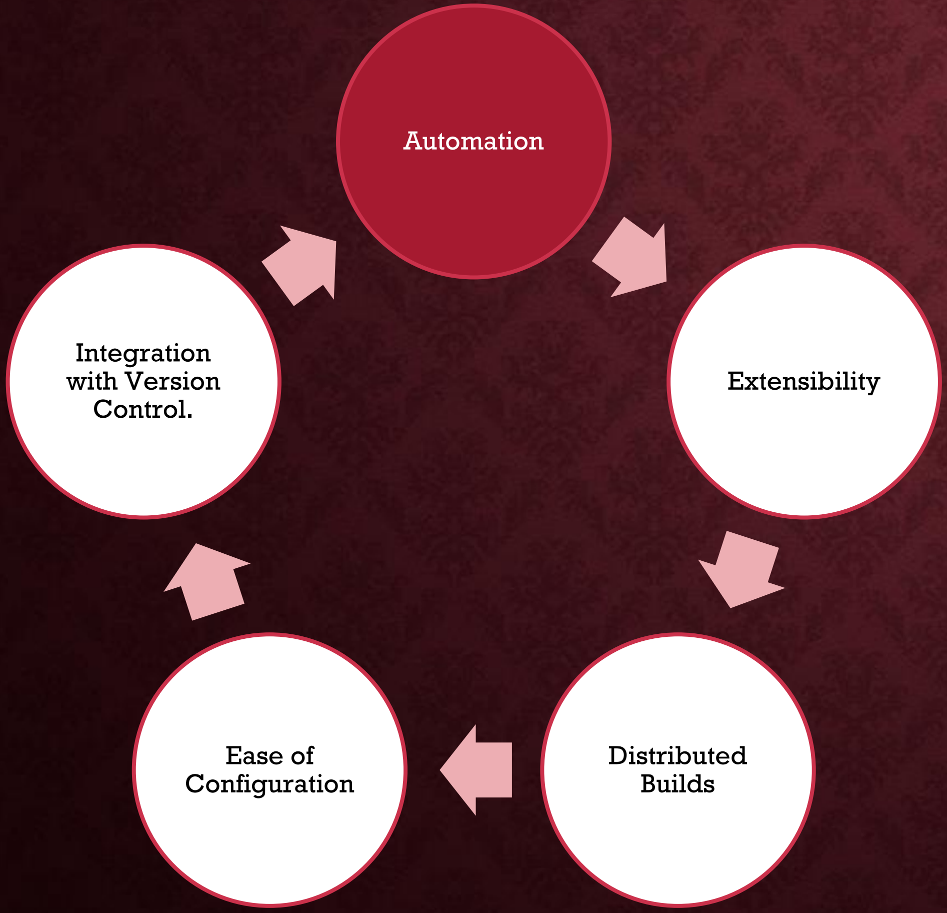
Extensibility

Distributed Builds

Ease of Configuration

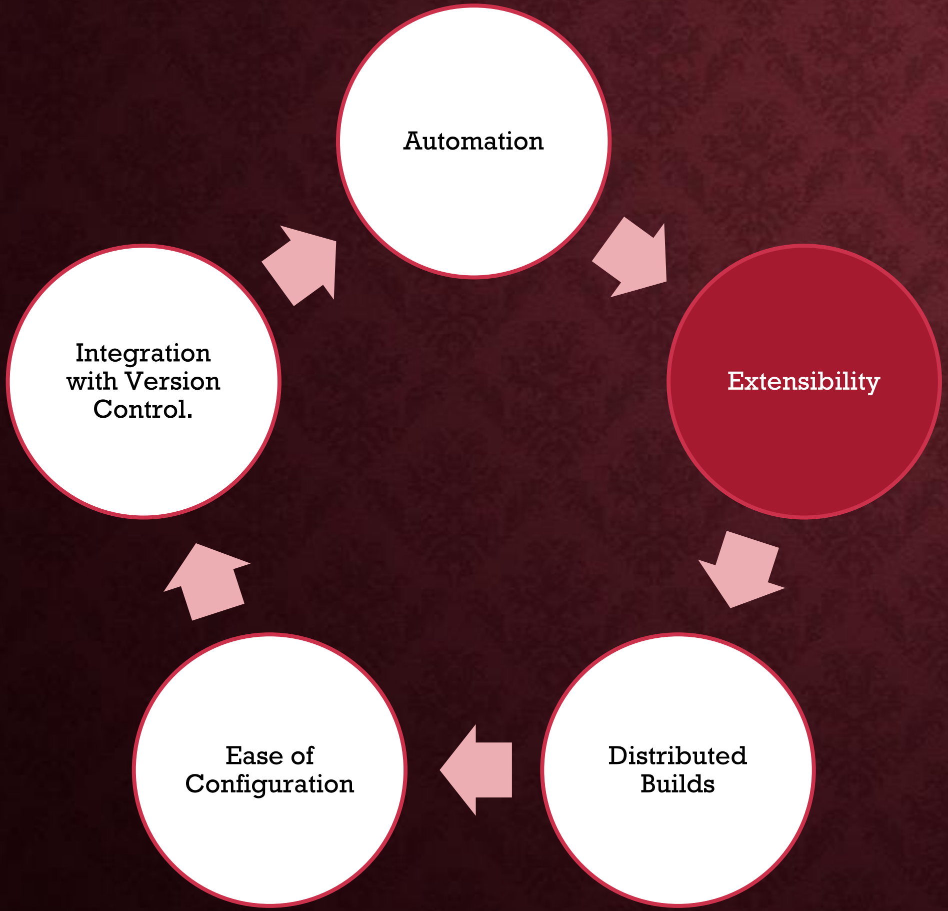
Integration with Version Control.

KEY FEATURES OF JENKINS



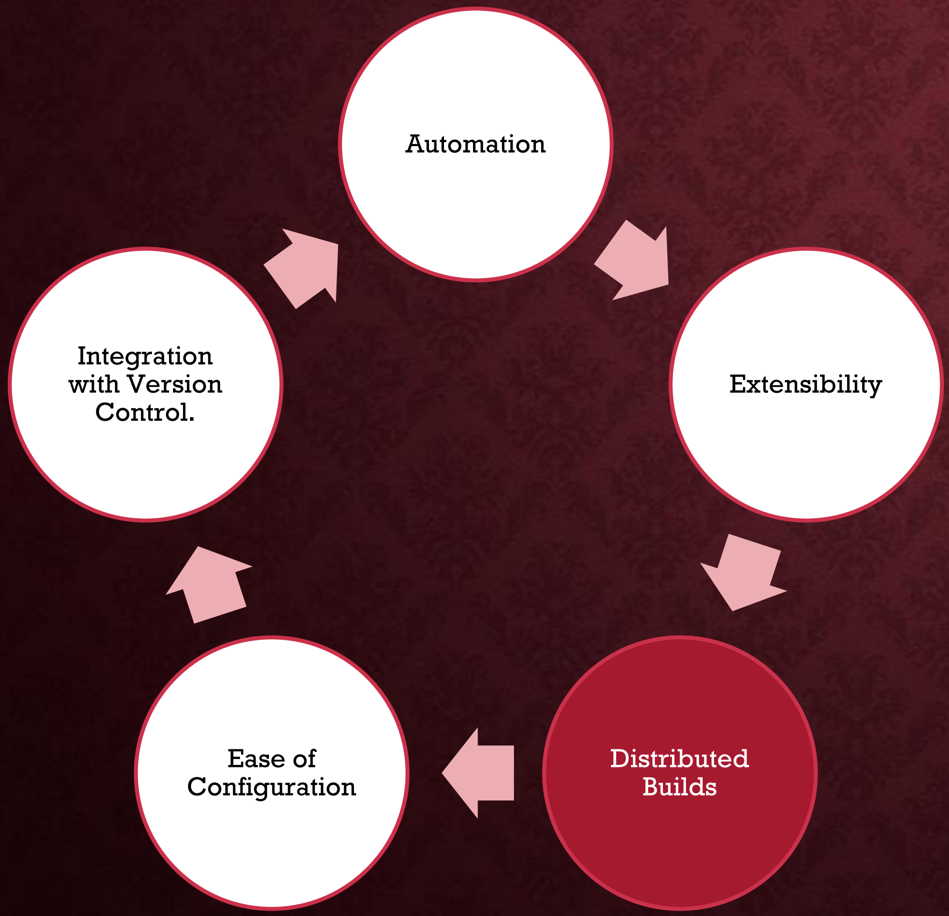
Jenkins automates various tasks, including building, testing, and deploying applications, reducing manual intervention and errors.

KEY FEATURES OF JENKINS



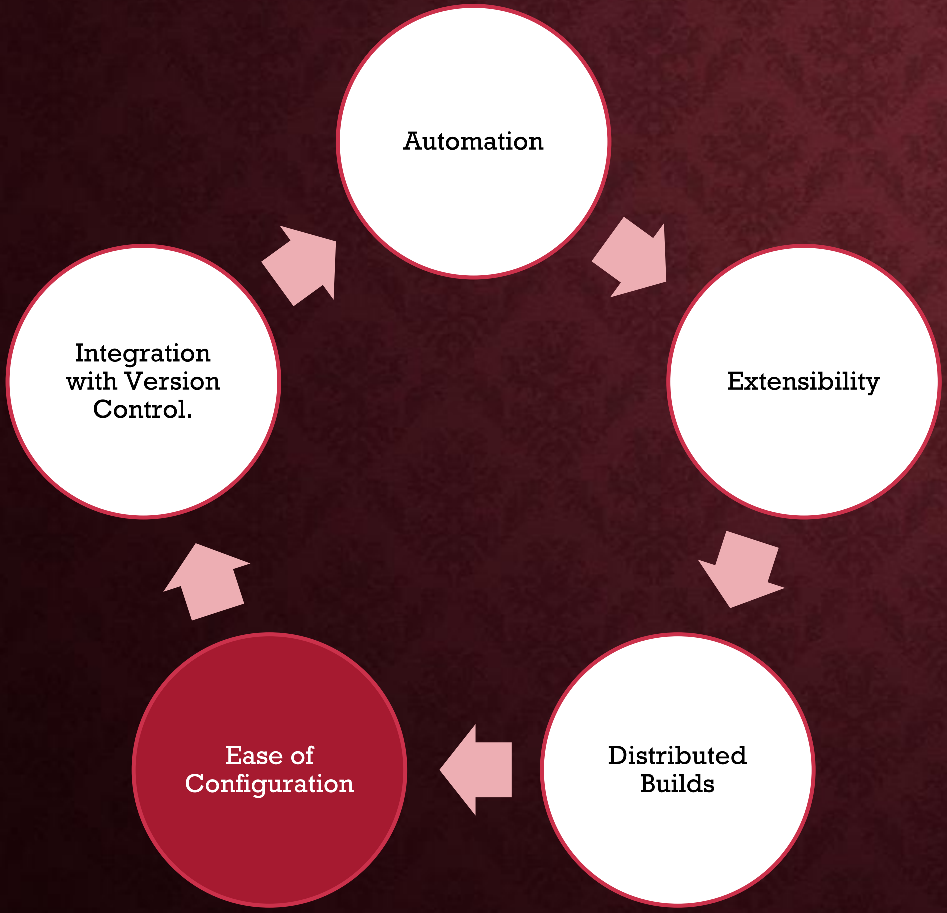
Jenkins supports a vast array of plugins that extend its functionality, allowing integration with different tools and technologies.

KEY FEATURES OF JENKINS



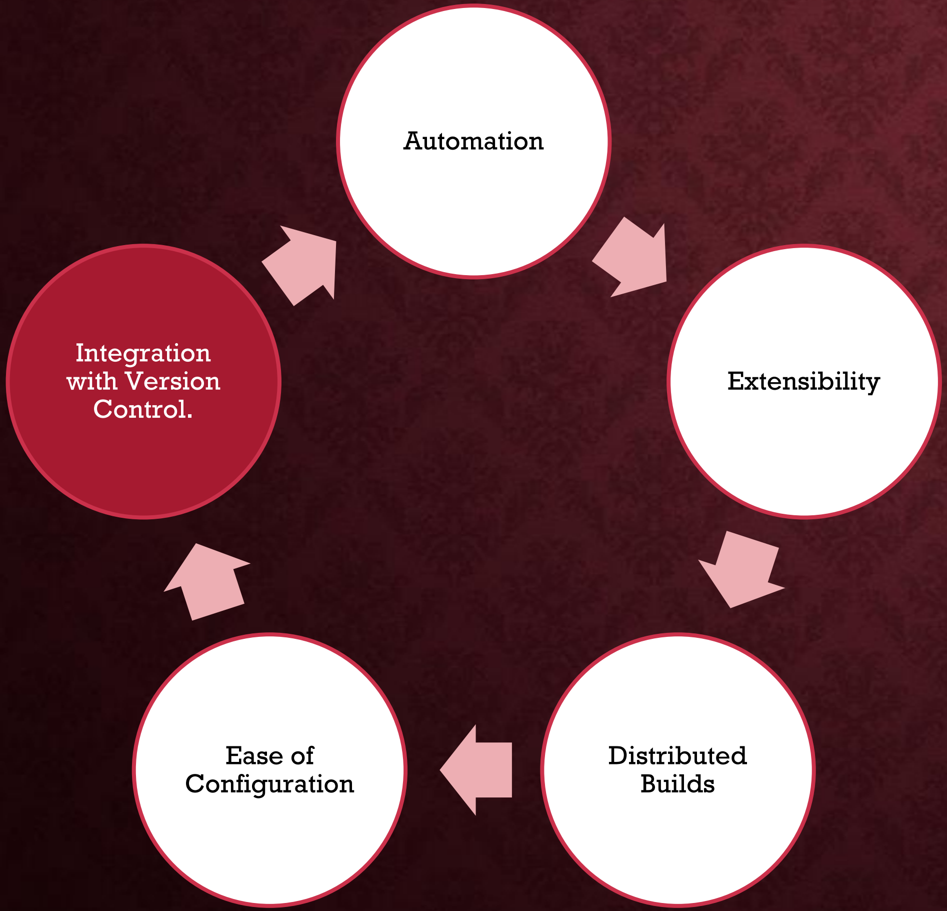
Jenkins supports distributed builds, enabling the use of multiple machines to execute tasks concurrently and speed up the build process.

KEY FEATURES OF JENKINS



Jenkins is configured through a web interface, making it accessible and user-friendly for both developers and administrators.

KEY FEATURES OF JENKINS



Jenkins seamlessly integrates with version control systems like Git, enabling continuous integration with code repositories.

CI/CD WITH JENKINS

Continuous Integration (CI):

- Jenkins ensures that code changes made by developers are automatically integrated into a shared repository.
- It involves building and testing the application every time new code is committed.

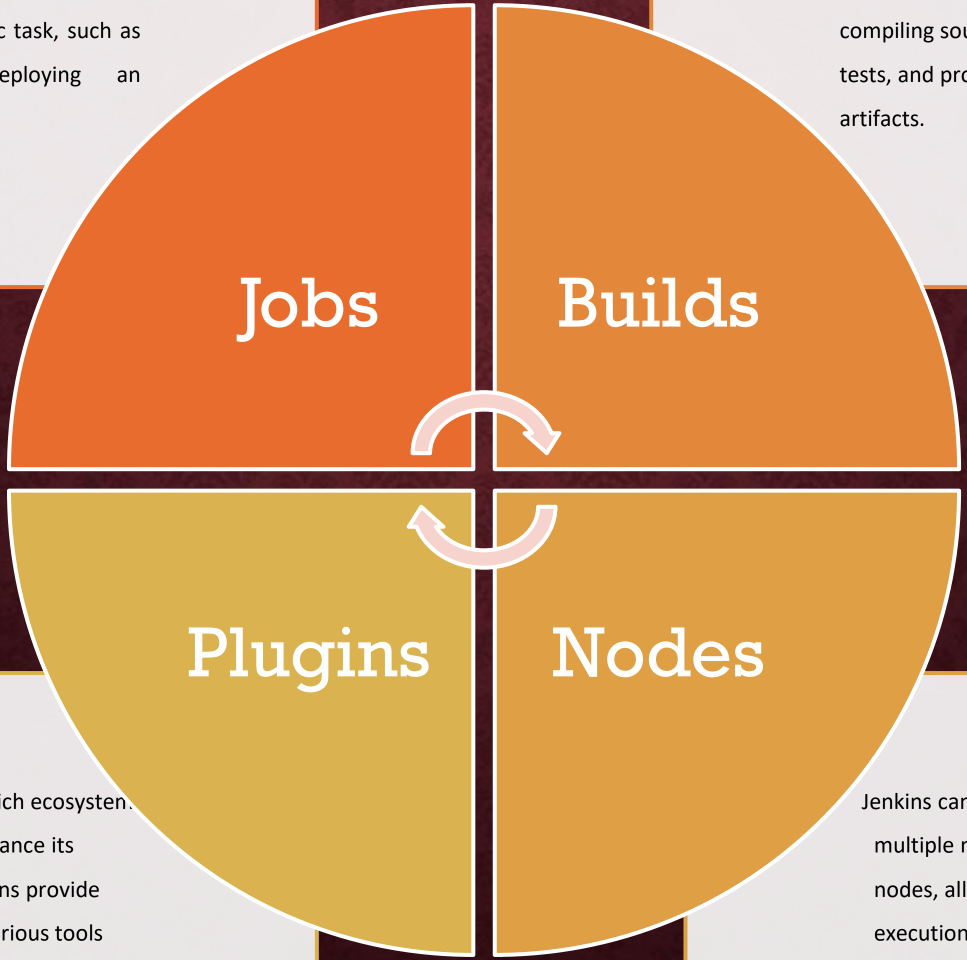
Continuous Delivery (CD):

- Jenkins extends CI to CD by automating the deployment process. It can deploy applications to staging or production environments based on predefined conditions.

CONCEPT

A job is a unit of work that performs a specific task, such as building or deploying an application.

It refers to the process of compiling source code, running tests, and producing executable artifacts.



Jobs

Builds

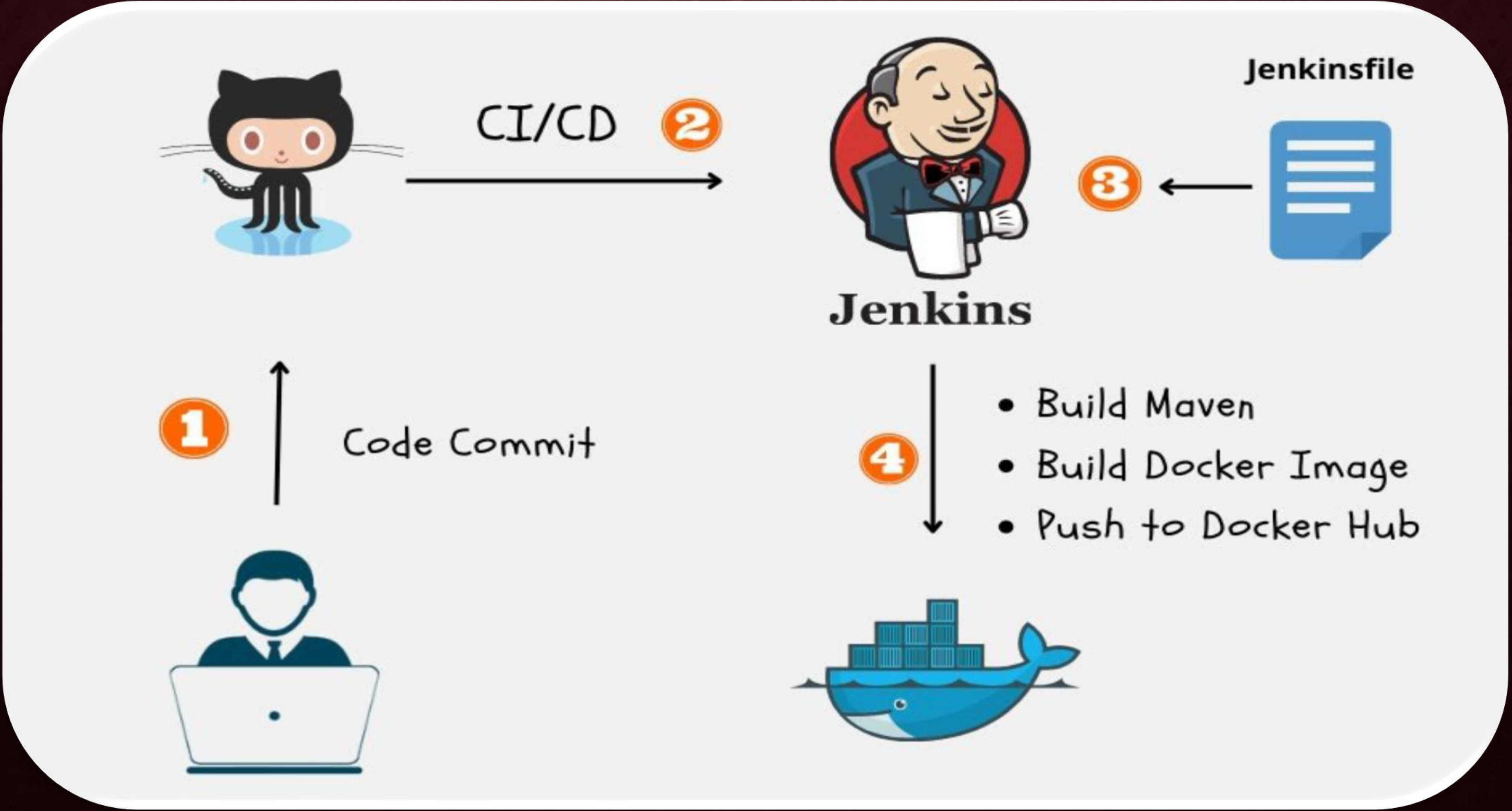
Plugins

Nodes

Jenkins supports a rich ecosystem of plugins that enhance its functionality. Plugins provide integration with various tools and services.

Jenkins can distribute work across multiple machines known as nodes, allowing parallel execution of tasks

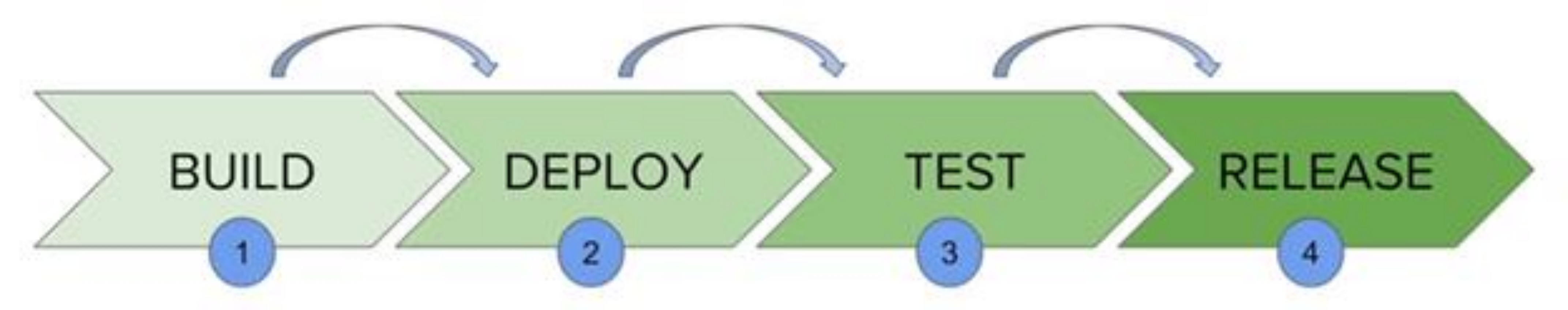
JENKINS ARCHITECTURE



JENKINS PIPELINE

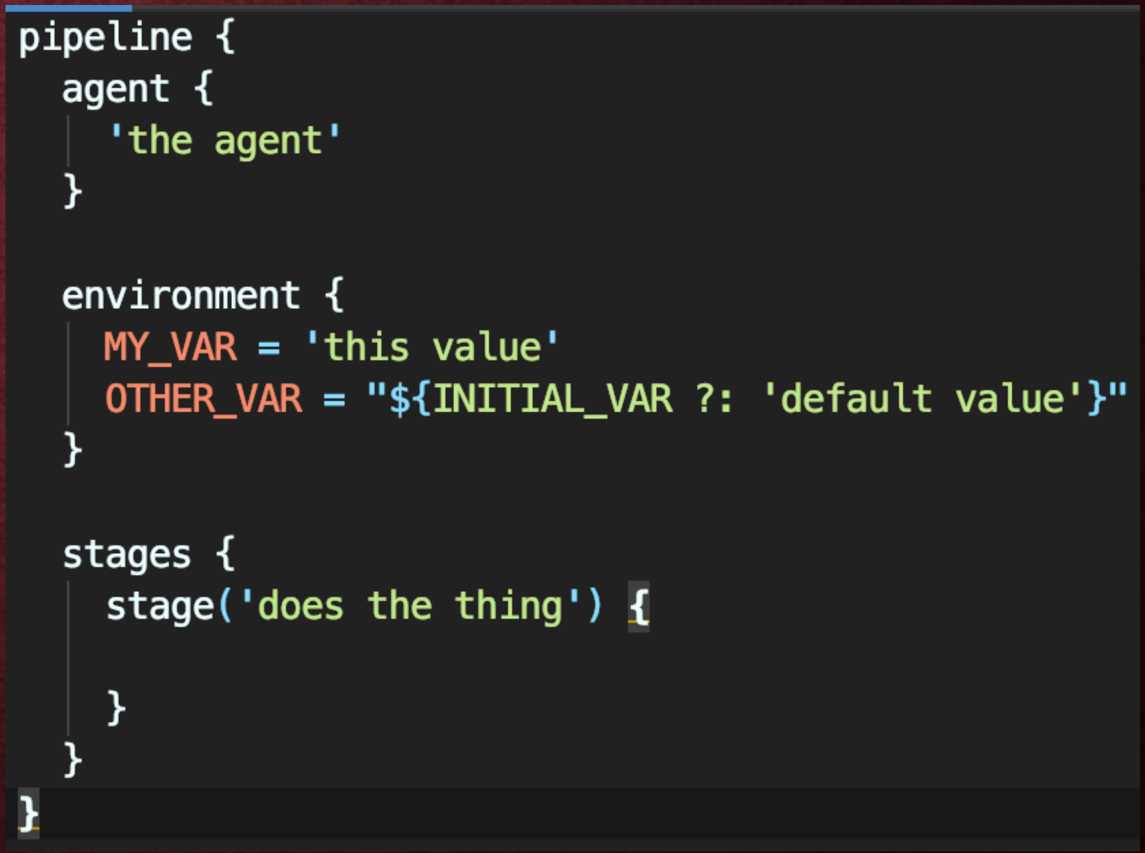
- A pipeline is a collection of events or jobs which are interlinked with one another in a sequence.
- A Jenkins Pipeline is a collection of jobs or events that brings the software from version control into the hands of the end users by using automation tools.
- It is used to incorporate continuous delivery in our software development workflow.
- A pipeline has an extensible automation server for creating simple or even complex delivery pipelines "as code", via DSL (Domain-specific language).

CONTINUOUS DELIVERY PIPELINE



JENKINS FILE

- Jenkins Pipeline can be defined by a text file called JenkinsFile.
- You can implement pipeline as code using JenkinsFile, and this can be defined by using a DSL (Domain Specific Language).
- With the help of JenkinsFile, you can write the steps required for running a Jenkins Pipeline.



```
pipeline {
  agent {
    'the agent'
  }

  environment {
    MY_VAR = 'this value'
    OTHER_VAR = "${INITIAL_VAR ?: 'default value'}"
  }

  stages {
    stage('does the thing') {

    }
  }
}
```

PIPELINE SYNTAX

Two types of syntax are used for defining your JenkinsFile.

1. Declarative

    - A simple way to create pipelines.
    - It consists of a predefined hierarchy to create Jenkins pipelines.
    - It provides you the ability to control all aspects of a pipeline execution in a simple, straightforward manner.

PIPELINE SYNTAX

Two types of syntax are used for defining your JenkinsFile.

2. Scripted

   - Scripted Jenkins pipeline syntax runs on the Jenkins master with the help of a lightweight executor.
   - It uses very few resources to convert the pipeline into atomic commands.

BENEFITS OF USING JENKINSFILE

- You can make pipelines automatically for all branches and can execute pull requests with just one JenkinsFile.
- You can review your code on the pipeline.
- You can review your Jenkins pipeline.
- This is the singular source for your pipeline and can be customized by multiple users.
- JenkinsFile can be defined by using either Web UI or with a JenkinsFile.