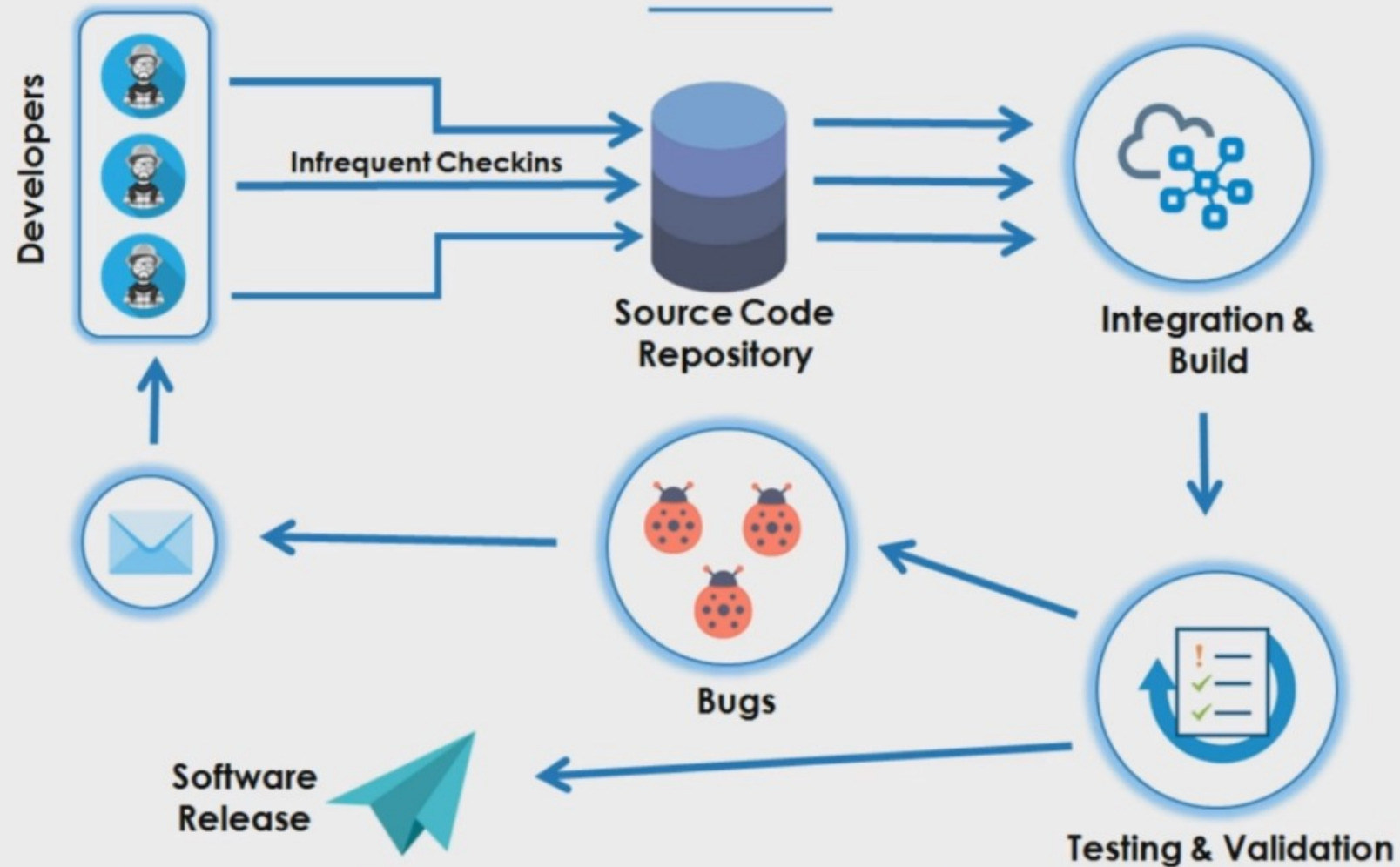


# Introduction to CI/CD with Jenkins

Rajeev gupta  
[rgupta.mtech@gmail.com](mailto:rgupta.mtech@gmail.com)  
Java Trainer & Consultant

# Process Before Continuous Integration



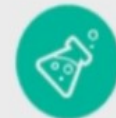
# Drawbacks of the Process

---



## No Constant Feedback

Developers have to wait till the application is deployed & the QA team verifies.



## Bug Fixing - Costly

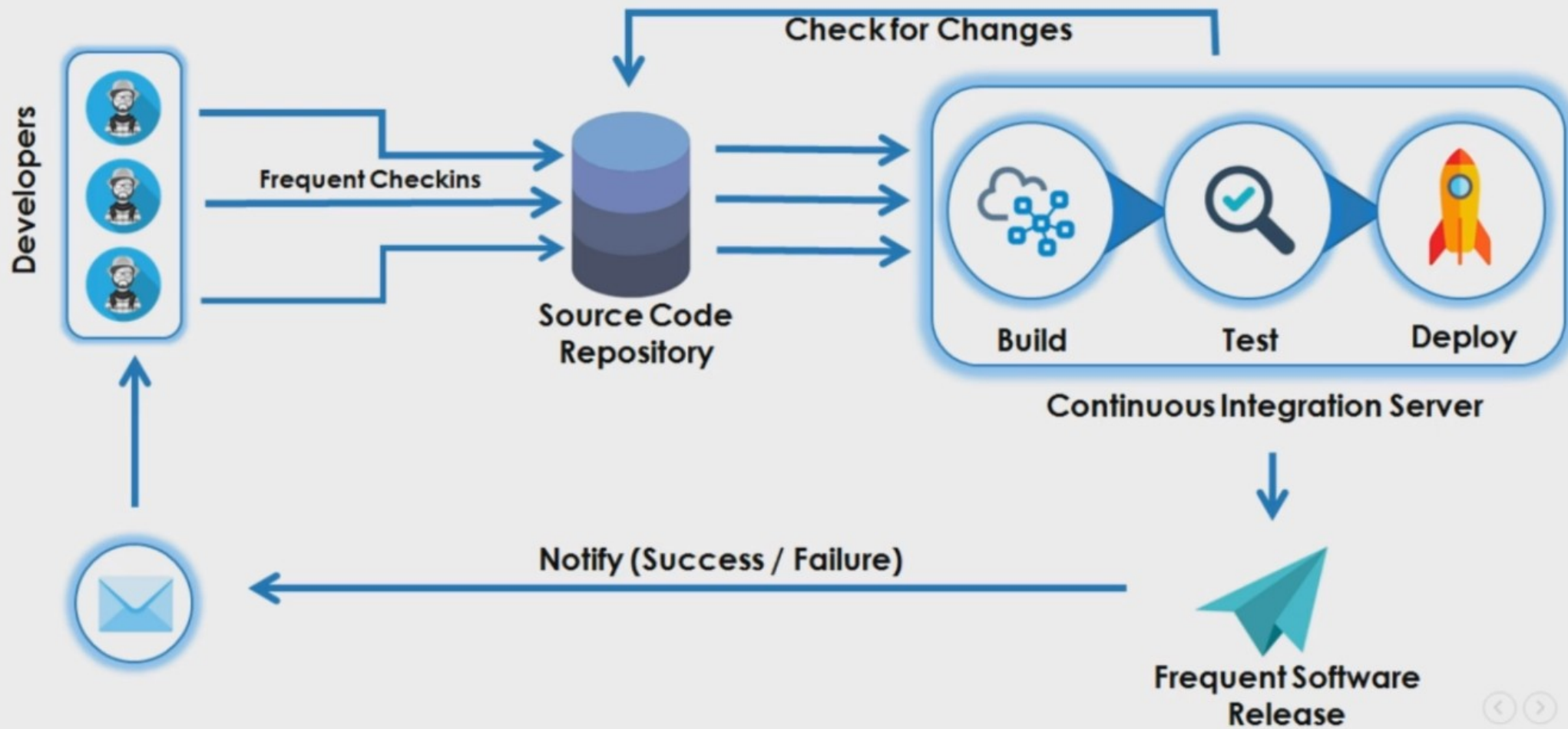
Locating & fixing the bugs by itself becomes a time consuming process.



## Slow Software Delivery

Software delivery process became slow & inconsistent with a lot of ifs & buts.

# Continuous Integration to the Rescue



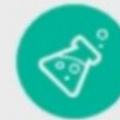
# Benefits of the Process

---



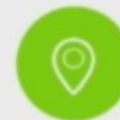
## Constant Feedback

Developers get constant feedback because of the automated build & test process.



## Bug Fixing - Effective

Time taken to identify the cause of the bug is reduced drastically.



## Fast Software Delivery

The frequency of the software releases increases.

# What is Continuous Integration?

---

## What is Continuous Integration?



**Continuous Integration (CI)** is a development practice where developers integrate code into a shared repository frequently, preferably several times a day. Each integration can then be verified by an automated build and automated tests.

## Key Benefits Include



One of the key benefits of integrating regularly is that you can detect errors quickly and locate them more easily.

## Continuous Delivery & Deployment



Continuous Deployment and Continuous Delivery have developed as best-practices for keeping your application deployable at any point



# CI CD



What is the difference between Continuous Integration, Continuous Deployment & Continuous Delivery?

1

## Continuous Integration (CI)

This is the practice of integrating changes from different developers in the team into a mainline as early as possible, in best cases several times a day. Each integration can then be verified by an automated build and automated tests.

2

## Continuous Delivery (CD)

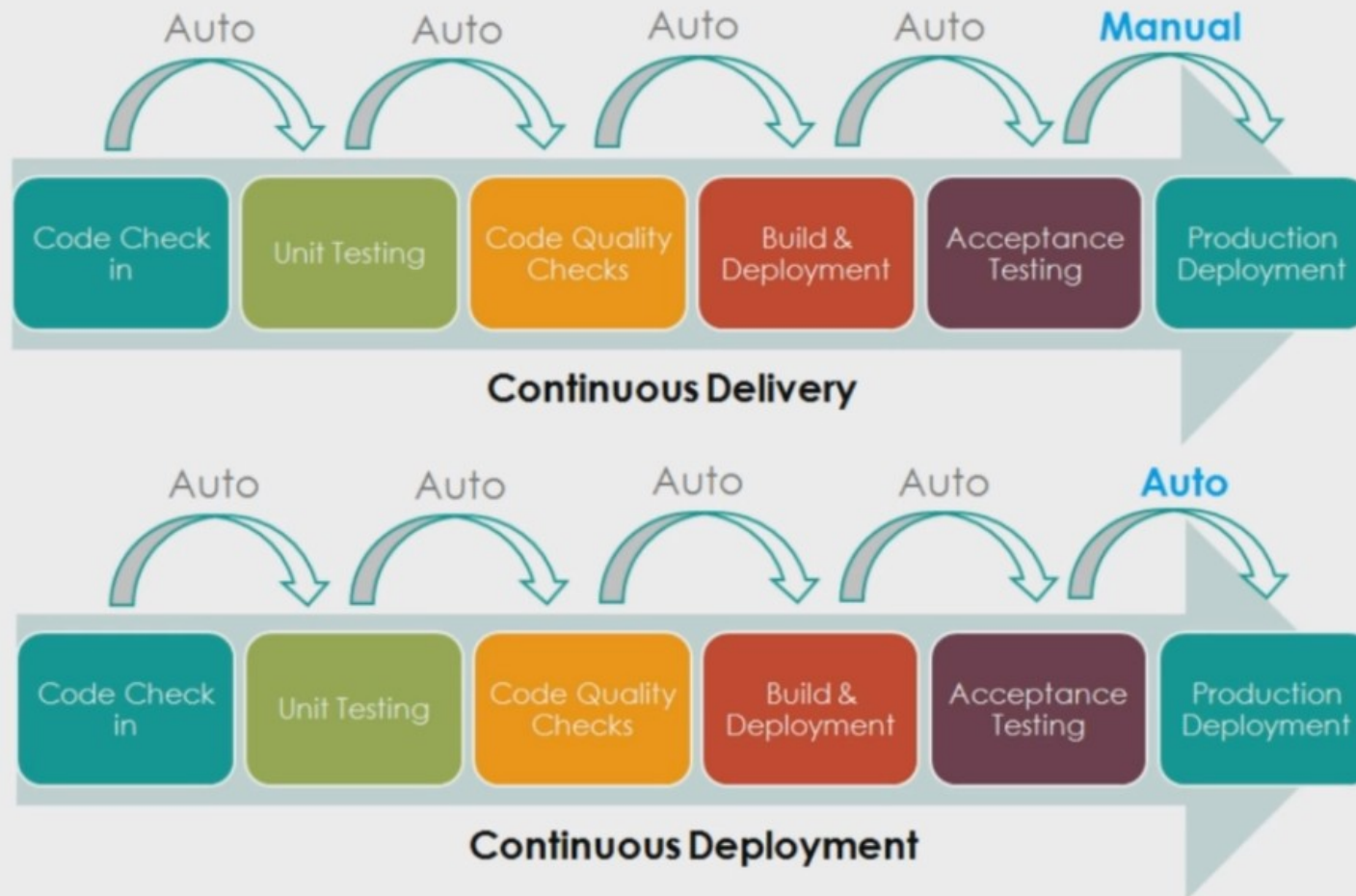
This is the practice of keeping your codebase deployable at any point. Beyond making sure your application passes automated tests it has to have all the configuration necessary to push it into production.

3

## Continuous Deployment

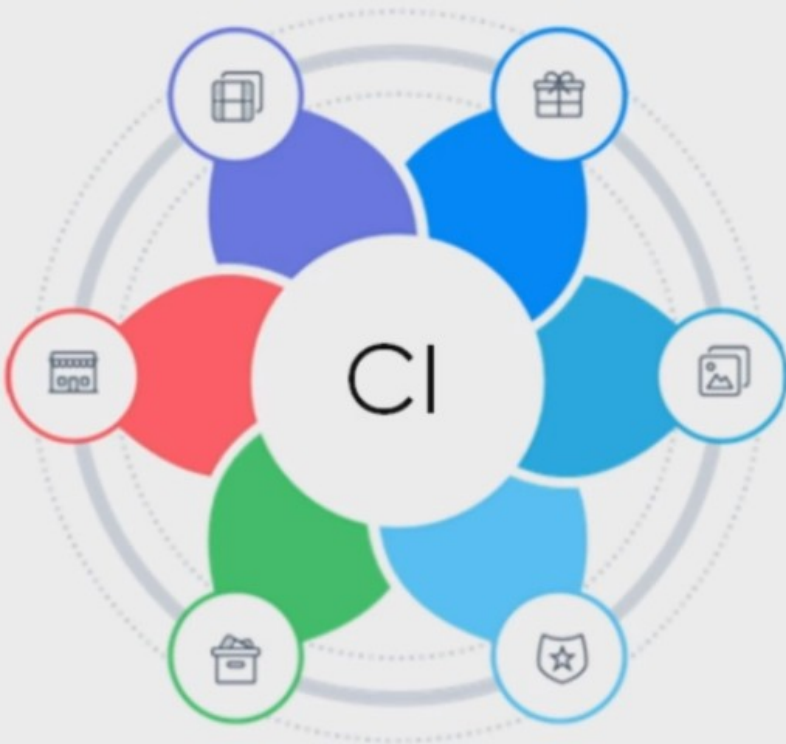
This is closely related to Continuous Integration and refers to keeping your application deployable at any point or even automatically releasing to a test or production environment if the latest version passes all automated tests

# Continuous Delivery Vs Deployment





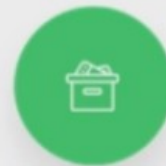
# Benefits of Continuous Integration



Reduces Risk



Better Communication



Faster Iterations



Reduces Overhead



Earlier Detection



Helps Collaboration

# CI – Principles & Practices

## The Principles

- ❑ Maintain a single source repository
- ❑ Automate the build
- ❑ Make your build self-testing
- ❑ Every commit should build on an integration machine
- ❑ Keep the build fast
- ❑ Test in a clone of the production environment
- ❑ Make it easy for anyone to get the latest executable version
- ❑ Everyone can see what's happening
- ❑ Automate deployment

## The Practices

- ❑ Developers check out code into their private workspaces
- ❑ When done, commit the changes to the repository
- ❑ The CI server monitors the repository and checks out changes when they occur
- ❑ The CI server builds the system and runs unit and integration tests
- ❑ The CI server releases deployable artefacts for testing
- ❑ The CI server assigns a build label to the version of the code it just built
- ❑ The CI server informs the team of the successful build
- ❑ If the build or tests fail, the CI server alerts the team
- ❑ The team fixes the issue at the earliest opportunity
- ❑ Continue to continually integrate and test throughout the project

# What is Jenkins?

## What is Jenkins?



**Jenkins** is the most popular open source continuous integration & continuous delivery server capable of orchestrating a chain of events / actions that help to achieved CI in an automated fashion.

## Across Complete Lifecycle



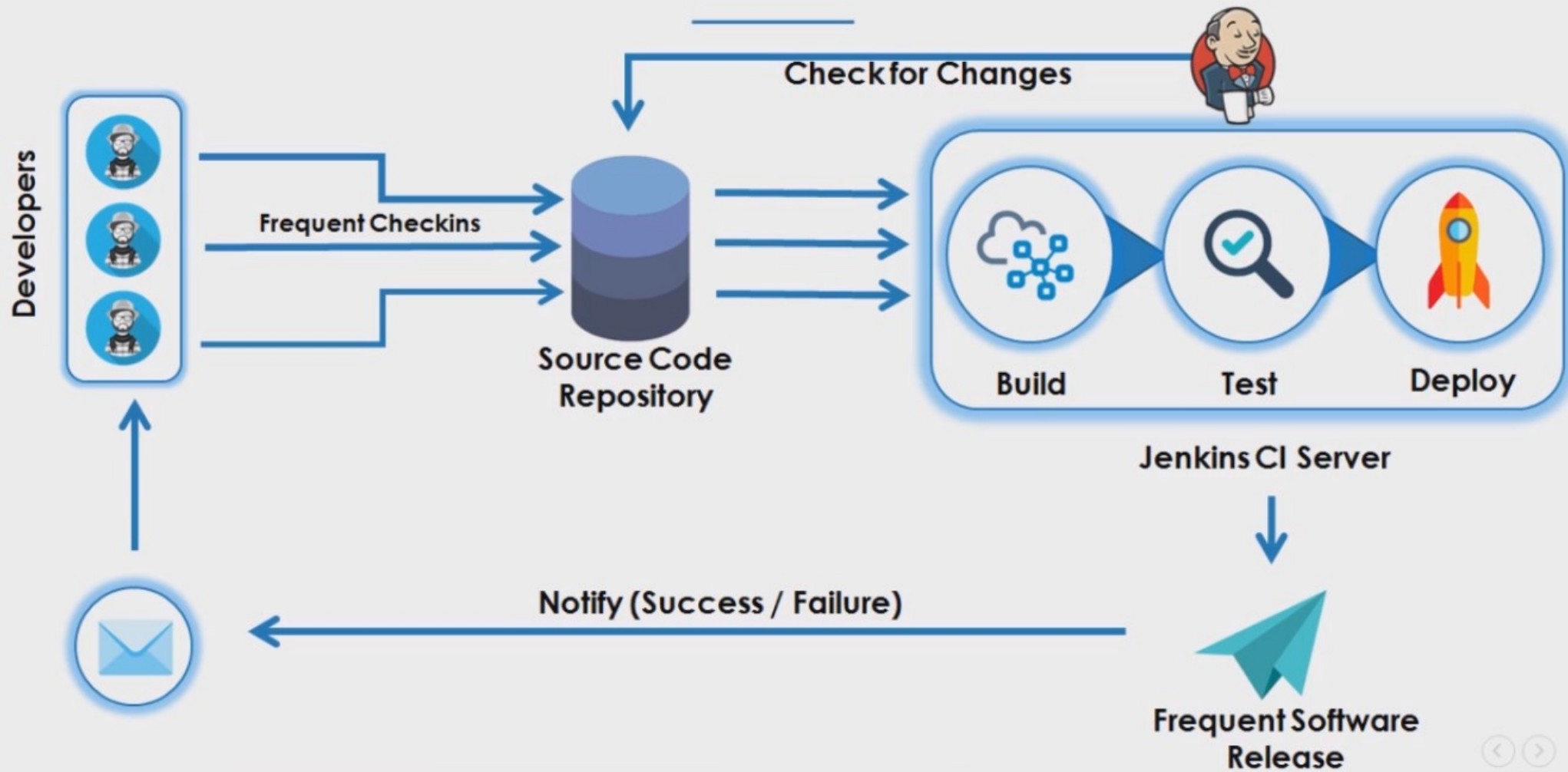
Jenkins supports the complete development lifecycle of software from building, testing, documenting the software, deploying and other stages of a software development lifecycle.

## Jenkins Plugins



Jenkins Plugins are at the heart of how Jenkins functions in achieving CI/CD. Jenkins is interconnected with well over 1,000 plugins that allow it to integrate with most of the development, testing and deployment tools.  
Refer - <https://plugins.jenkins.io/>

# CICD with Jenkins



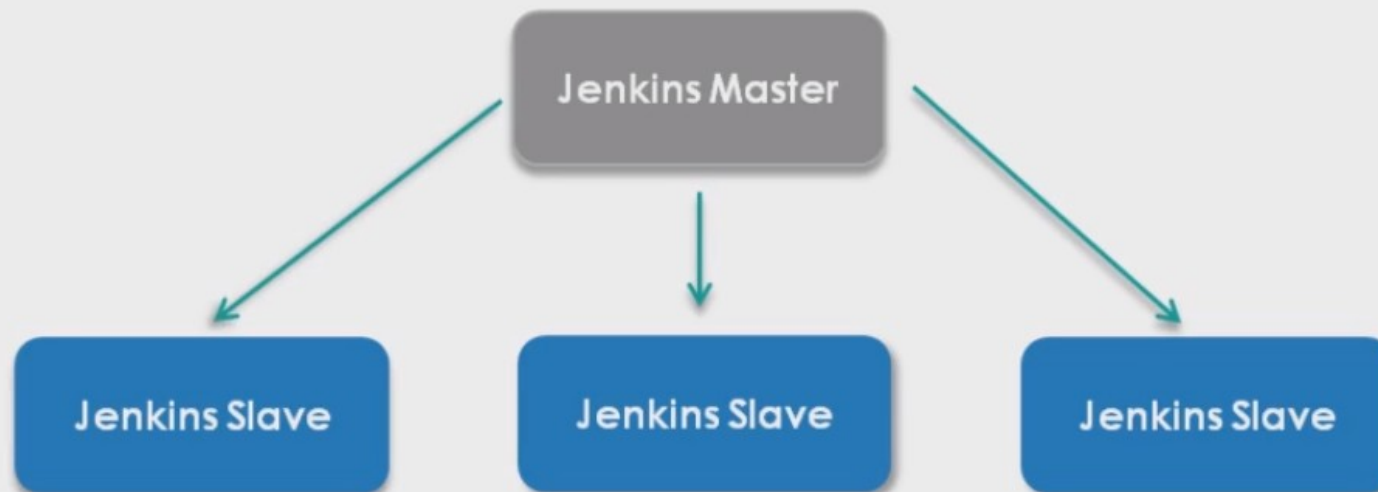
# Features of Jenkins





# Jenkins Distributed Architecture

Scheduling & Dispatching jobs to slaves, Monitor the slaves, Record & present job results, execute some jobs directly



Handles all requests from the Jenkins Master instance. Each slave is an independent entity running on its own OS.



# Jenkins Pipeline

Average stage times:  
(Average full run time: ~5s)

	build	test: integration-& quality	test: functional	test: load-& security	approval	deploy: prod
#17 Sep 22 15:05 No Changes ⌂ Retry ⬇ Download	836ms	20min 43s	9ms	7ms	89ms	5ms
#16 Sep 22 15:04 No Changes ⌂ Retry ⬇ Download	538ms	10s	10ms	8ms	72ms (queued for 7s)	4ms
#15 Sep 22 15:03 No Changes ⌂ Retry ⬇ Download	479ms	6s	9ms	9ms	74ms (queued for 8s)	5ms
#14 Sep 22 15:03 No Changes ⌂ Retry ⬇ Download	922ms	6s	10ms	9ms failed		
#13 Sep 22 15:02 No Changes ⬇ Download	1s	8s	12ms	9ms	80ms (queued for 5s)	5ms
#12 Sep 22 15:02 No Changes ⌂ Retry ⬇ Download	942ms	9s	13ms failed			
	1s	6s	13ms	11ms	111ms (queued for 5s)	absorbed

**Jenkins Pipeline** (or simply "Pipeline") is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.

A continuous delivery pipeline is an automated expression of your process for getting software from version control right through to your users and customers.

# Developer & Build Admin Roles

---

## Developer Roles

- Coding
- Unit Testing
- Code Review
- Push code to Repository

## Build Admin Roles

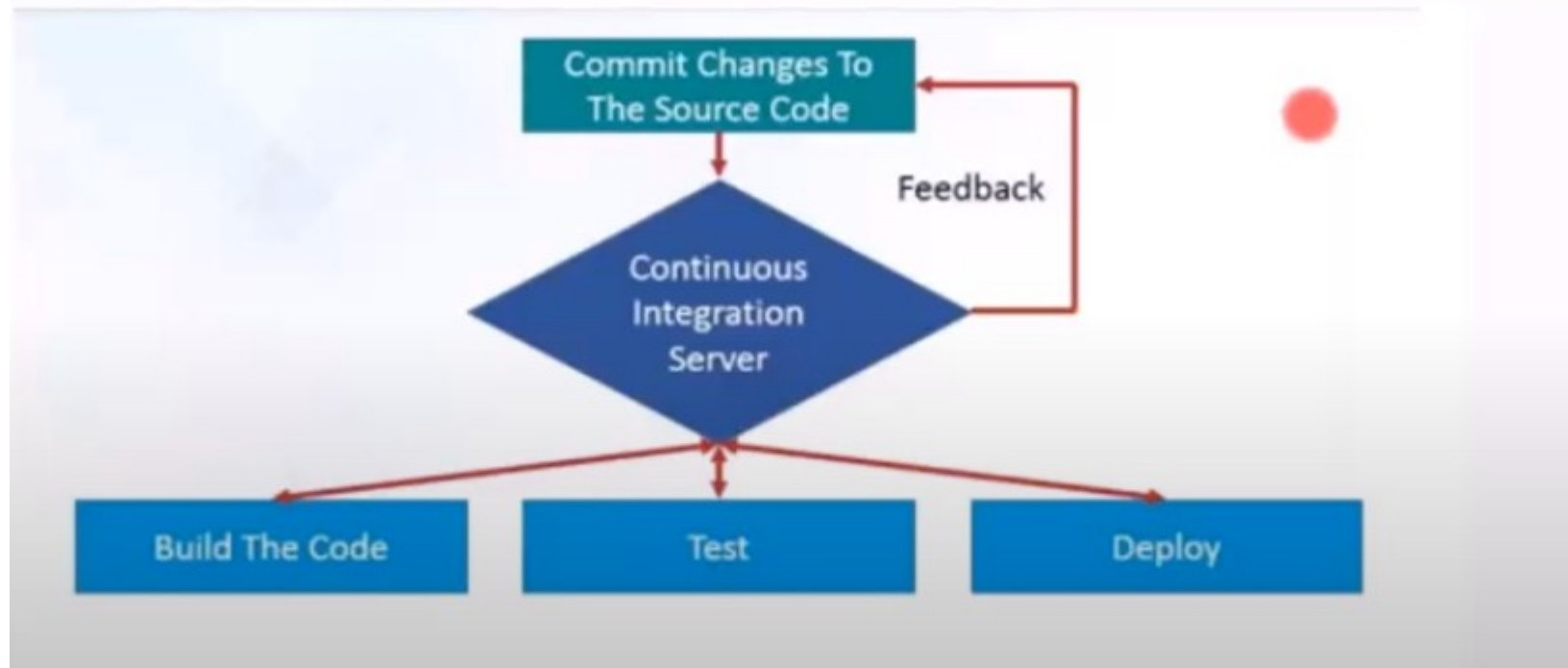
- Pull Latest Code from Repo
- Compile Source Code
- Execute Unit Test Suites
- Package source code as jar/war/ear
- Deploy to Server

# Dis Advantages of Manual Build Process

---

- Dedicated Administrators are required to do project builds
- Developers needs to send build requests to admin to do the deployment when new committed to repo.
- Admin needs to spend some to pull the code, to compile and to package and to deploy.
- It is time consuming process.

# CI Work Flow



# CI Benefits

---

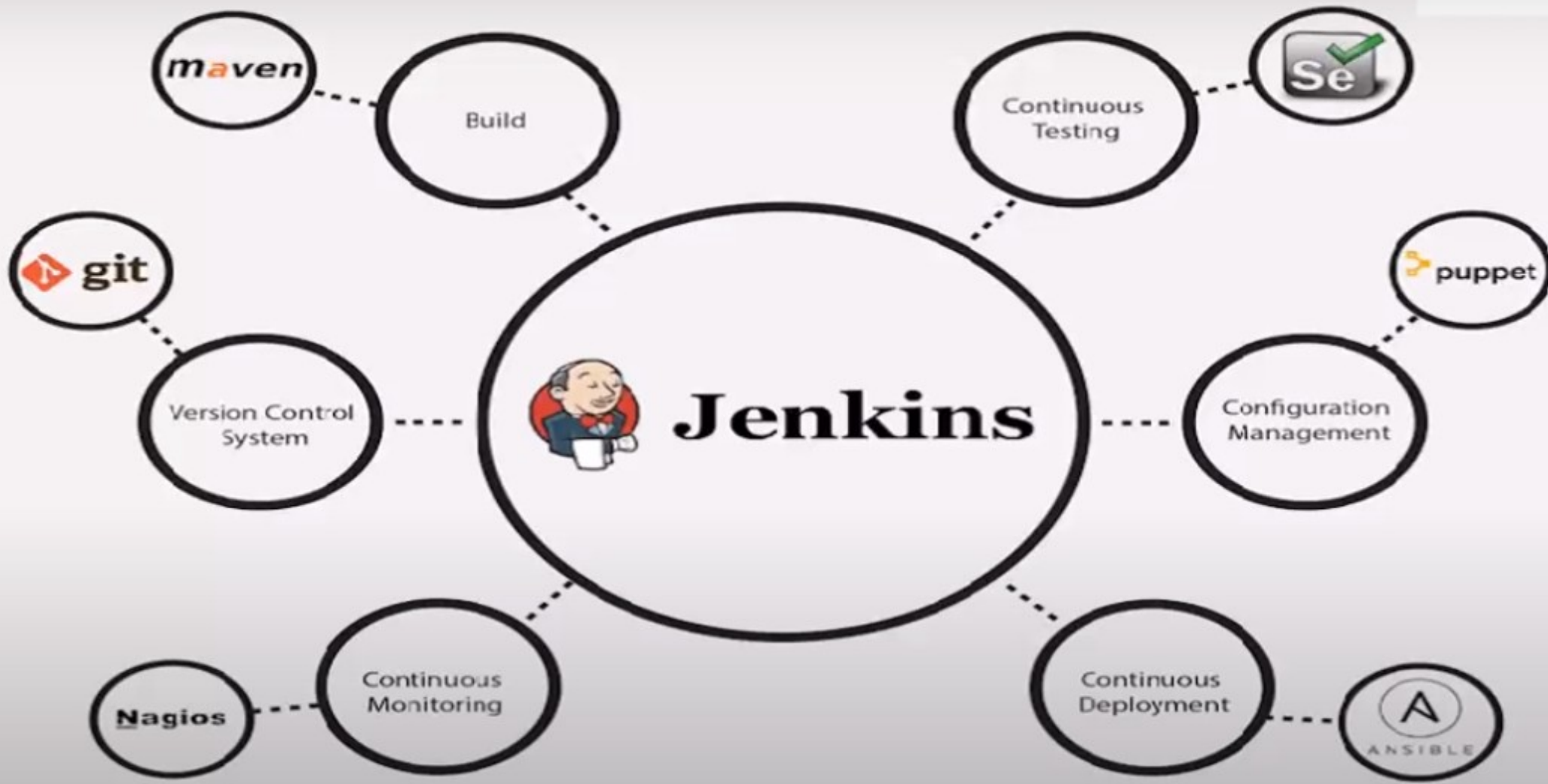
- Immediate bug detection
- No integration step in the lifecycle
- A deployable system at any given point
- Record of evolution of the project

# CI Tools

---

- **Code Repositories**
  - SVN, Mercurial, Git
- **Continuous Build Systems**
  - Jenkins, Bamboo, Cruise Control
- **Test Frameworks**
  - JUnit, Cucumber, CppUnit





# What is Jenkins

---

- Jenkins is the leading open source continuous integration tool.
- Jenkins was originally developed as the Hudson project. Hudson's creation started in summer of 2004 at Sun Microsystems.
- In November 2010, an issue arose in the Hudson community with respect to the infrastructure used. Negotiations were held between the principal project contributors and Oracle; a key sticking point was the control of the name "Hudson" itself, which Oracle claimed, and on January 11, 2011, a proposal was made to change the project name from "Hudson" to "Jenkins"

# Installing and Configuring Jenkins

---

- Download Jenkins: Open the official website of Jenkins :

<https://jenkins.io/index.html>

- After downloading, from the command prompt, browse to the directory where the jenkins.war file is present. Run the following command:

```
java -jar jenkins.war
```

- Extraction of the war file is done by an embedded webserver called winstone.
- Jenkins by default runs on port 8080, but we can run it on different port by issuing following command: `java -jar jenkins.war --httpPort=8181`

# Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	** Resource Disposer
✓ Timestamper	✓ Workspace Cleanup	✓ Ant	✓ Gradle	** Matrix Project
⌚ Pipeline	⌚ GitHub Branch Source	⌚ Pipeline: GitHub Groovy Libraries	⌚ Pipeline: Stage View	Workspace Cleanup
⌚ Git	⌚ Subversion	⌚ SSH Slaves	⌚ Matrix Authorization Strategy	Ant
⌚ PAM Authentication	⌚ LDAP	⌚ Email Extension	⌚ Mailer	Workspace Cleanup
				Ant
				Gradle
				Gradle
				** Pipeline: Milestone Step
				** Pipeline: Milestone Step
				** JavaScript GUI Lib: jQuery bundles (jQuery and jQuery UI)
				** JavaScript GUI Lib: jQuery bundles (jQuery and jQuery UI)
				** Jackson 2 API
				** Jackson 2 API
				** JavaScript GUI Lib: ACE Editor

## Getting Started

# Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:



## Enter an item name

» Required field



### Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



### Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



### Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



### Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a



# Jenkins Job to Deploy Spring App in Tomcat Server

---

- Download and Install Tomcat Server
- Change Tomcat Server Port Number (If Required) in server.xml file
- Configure User with manager-script role in tomcat-user.xml file
- Start Tomcat server and Open Tomcat Server Admin Console
- Install Deploy Container Plugin in Jenkins
- Re-Start Jenkins Server to Reflect Container Plugin
- Create Jenkins Job to Deploy Application into Tomcat Server

# Jenkins Pipeline

---

- In Jenkins, a pipeline is a group of events or jobs which are interlinked with one another in a sequence.
- In simple words, Jenkins Pipeline is a combination of plugins that support the integration and implementation of continuous delivery pipelines using Jenkins

# Types of Pipelines

---

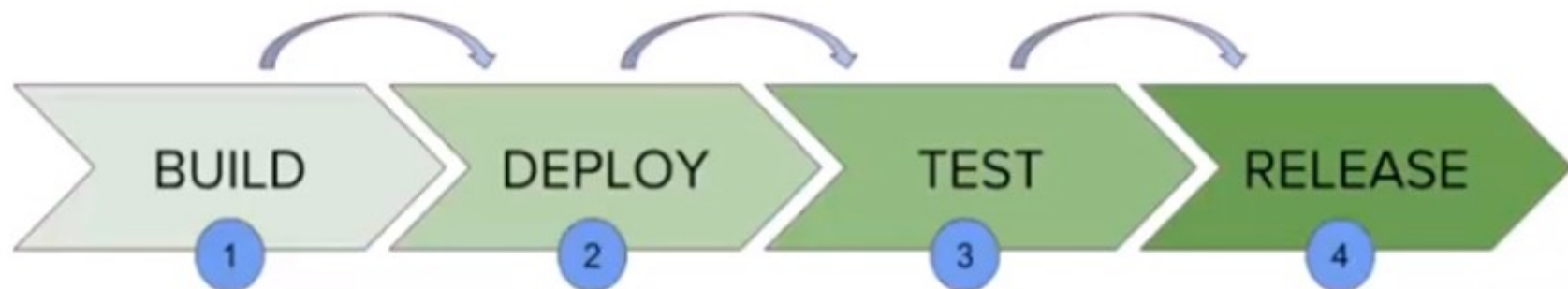
- **Build Plugin Pipeline**
- **Declarative Pipeline**
- **Scripted Pipeline**

# What is Continuous Delivery Pipelines?

## How it Works?

---

In a Jenkins pipeline, every job or event has some sort of dependency on at least one or more events.



# Install Build Pipeline Plugin in Jenkins

- With the build pipeline plugin, you can create a pipeline view of incoming and outgoing jobs, and create triggers which require manual intervention.
- Here is how you can install the build pipeline plugin in your Jenkins:

**Step 1)** The settings for the plugin can be found under **Manage Jenkins > Manage Plugins**.



# Pipeline Creation Overview

