



Week 12 : SOFTWARE DEVELOPMENT TOOLS AND ENVIRONMENTS

# Traditional Deployment



Developer



Operations

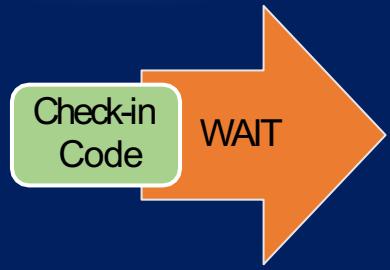
# Traditional Deployment



Developer



Operations



Hours/Days + Lot of Grief for Developer & Operations

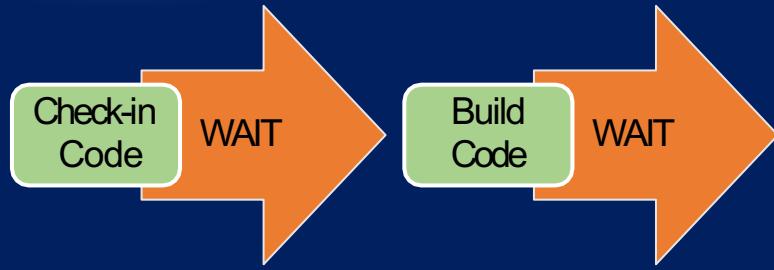
# Traditional Deployment



Developer



Operations



Hours/Days + Lot of Grief for Developer & Operations

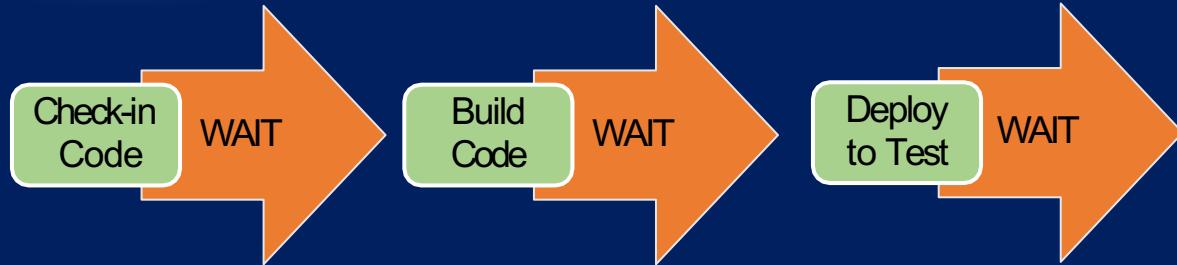
# Traditional Deployment



Developer



Operations



Hours/Days + Lot of Grief for Developer & Operations

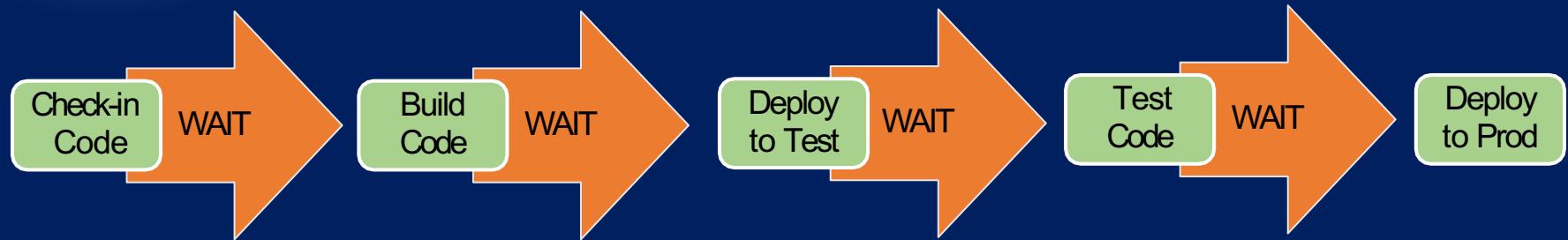
# Traditional Deployment



Developer



Operations



Hours/Days + Lot of Grief for Developer & Operations

# Traditional Deployment

When are  
you gonna  
deploy my  
code?

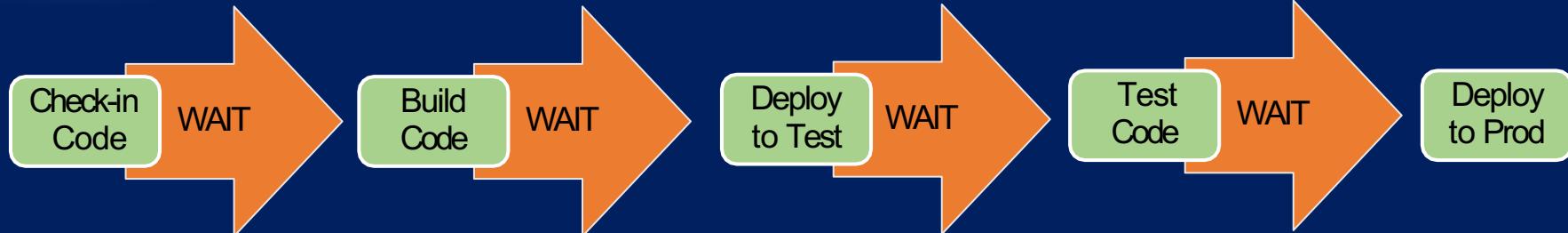


Developer

When you  
stop breaking  
my servers



Operations





I just  
wanna do  
cool stuff

SYSTEM

# What is DevOps?

- Word “DevOps” coined in 2009 by Patrick Debois
- Combination of cultural philosophies, practices, and tools
  - Job market is based on tools!
- Development and Operations teams are no longer “siloed”



# Traditional Deployment

When are  
you gonna  
deploy my  
code?

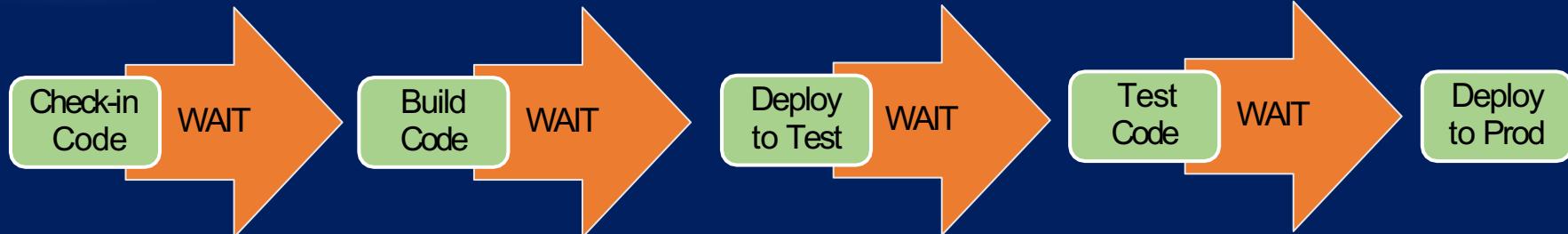


Developer

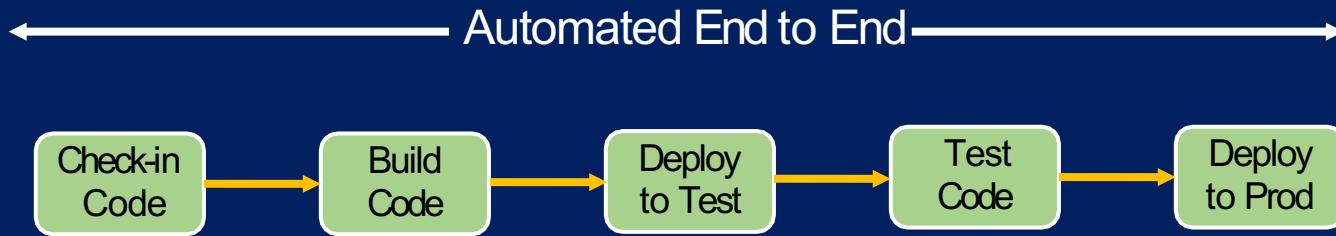
When you  
stop breaking  
my servers



Operations

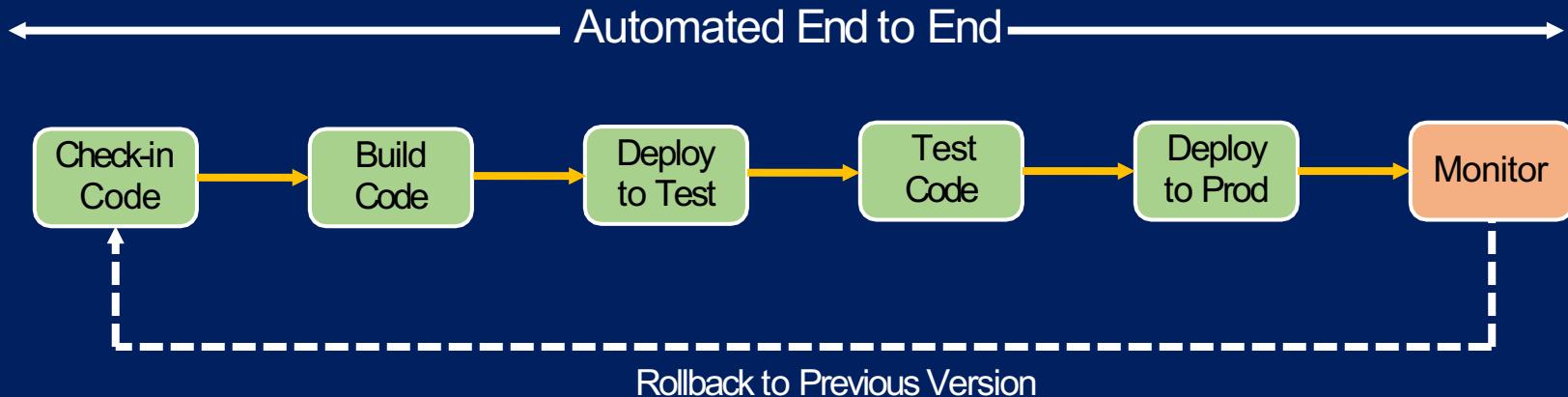


# Devops



- Whole flow done in seconds!
- Easy to rollback in case of errors

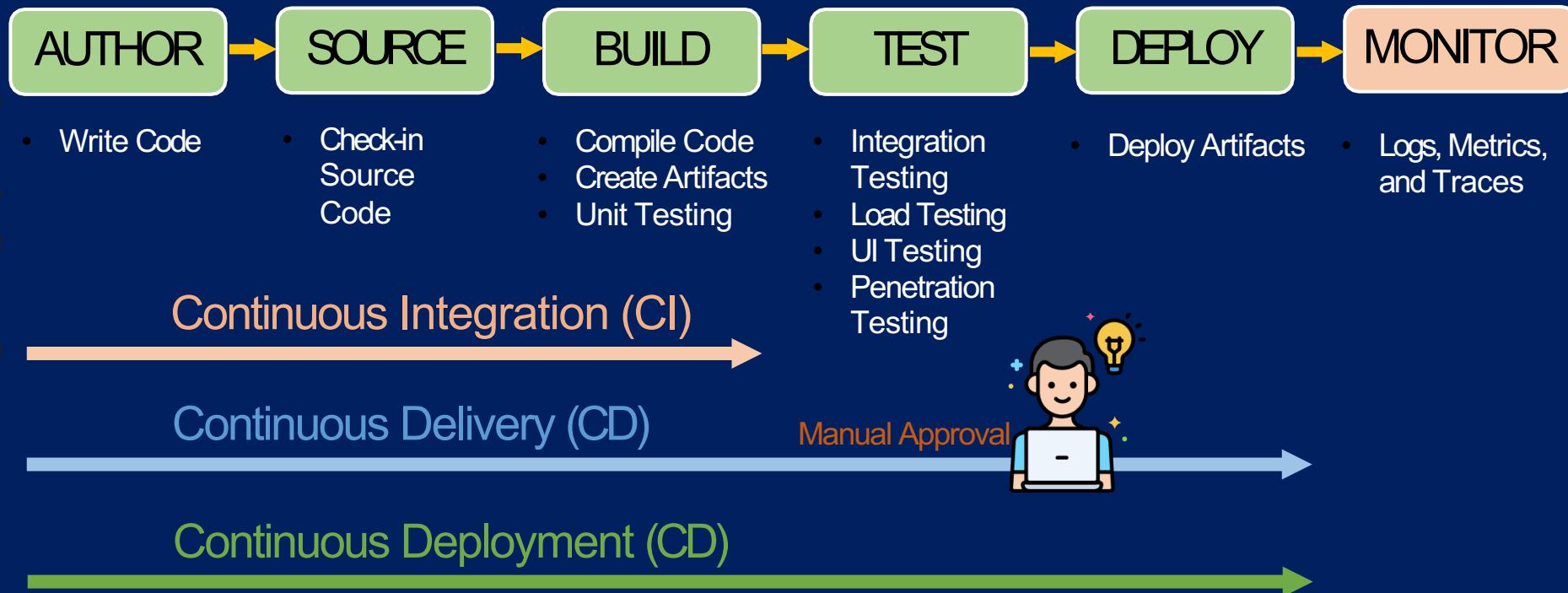
# DevOps



# General DevOps Practices

- Automate everything!
- Deploy frequently rather than one mega deployment in months
- Codify every step - infrastructure, application and more
- Rome was not built in a day!

# DevOps Phases



# DevOps Tools

JUnit



pytest



APACHE JMeter™



LOCUST



AWS Cloud9



AWS CodeCommit



AWS CodeBuild



AWS CodeBuild



AWS CodeBuild



AWS CodeBuild



AWS CodeDeploy



CloudWatch



AWS X-Ray



Prometheus



Grafana

AUTHOR

SOURCE

BUILD

TEST

DEPLOY

MONITOR

- Write Code
- Check-in Source Code
- Compile Code
- Create Artifacts
- Unit Testing
- Integration Testing
- Load Testing
- UI Testing
- Penetration Testing
- Deploy Artifacts
- Logs, Metrics, and Traces



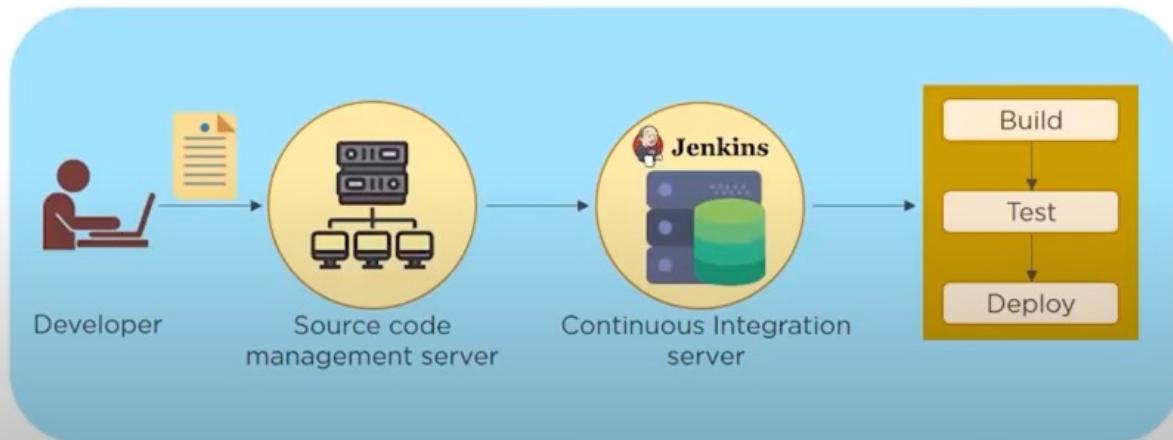
AWS CodePipeline



# What is Jenkins?

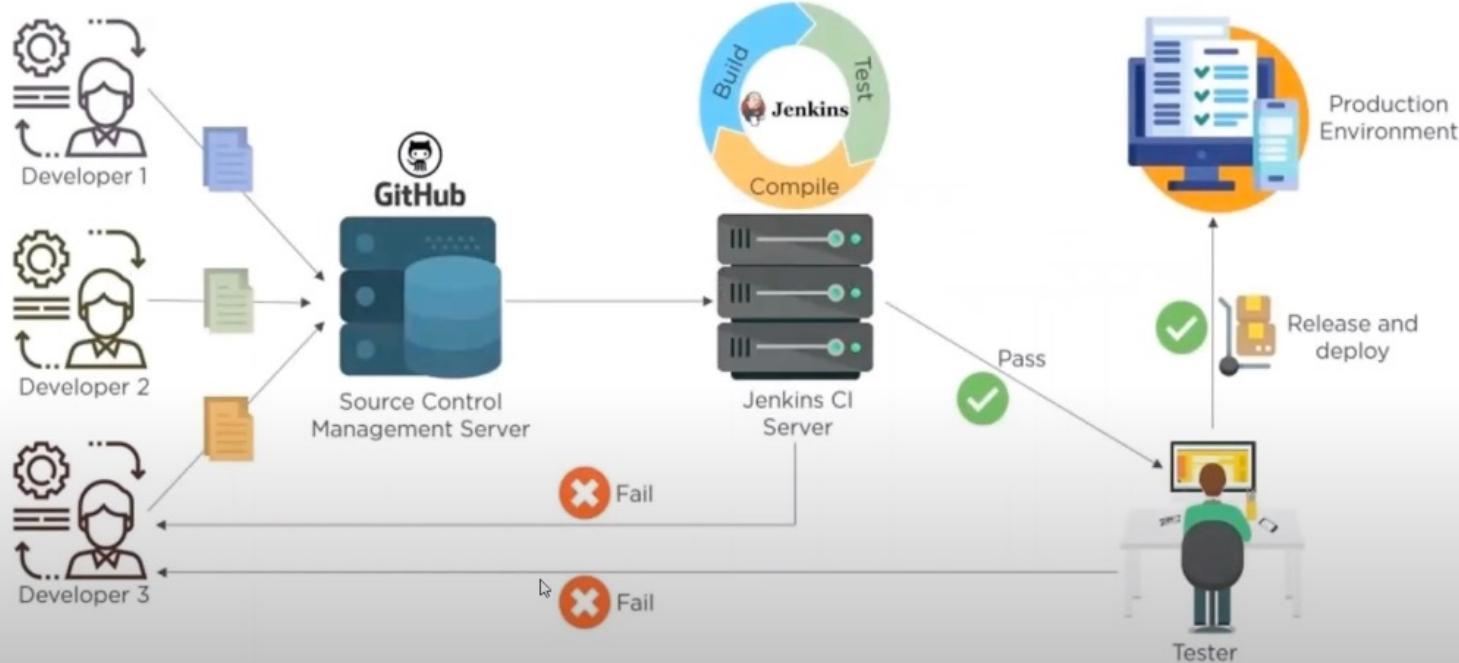


Jenkins is an open source Continuous Integration server written in Java that allows continuous development, test and deployment of codes



# CI & CD

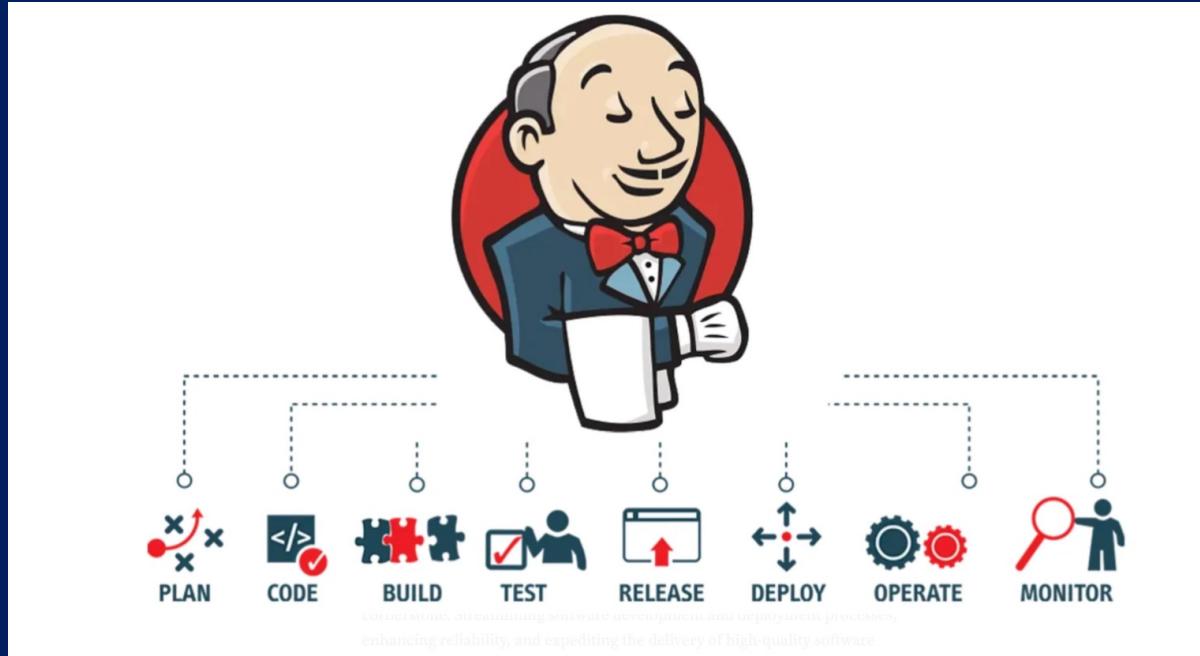
---



# Lab 01:

## 01\_1\_Jenkins Installation

## 01\_2\_Jenkins Plugin Installation



# 01\_1\_Jenkins Installation

## Installation Steps ↗

1. Update System Packages Update the list of available packages and their versions.

```
sudo apt-get update
```

2. Installation of Java

```
sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version
openjdk version "17.0.8" 2023-07-18
OpenJDK Runtime Environment (build 17.0.8+7-Debian-1deb12u1)
OpenJDK 64-Bit Server VM (build 17.0.8+7-Debian-1deb12u1, mixed mode, sharing)
```

3. Long Term Support release

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
```

4. Start Jenkins

```
sudo systemctl enable jenkins
```

```
sudo systemctl start jenkins
```

```
sudo systemctl status jenkins
```

# **LAB 02 : Jenkins Add Credentials**

# 1.Add SSH-Credentials

Jenkins

Dashboard >

+ New Item

People

Build History

Manage Jenkins

My Views

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job

Set up a distributed build

Set up an agent

Configure a cloud

Learn more about distributed builds

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

1

## Security

2

**Credentials**

Configure credentials

## Credential Providers

Configure the credential providers and types

## Status Information

**System Information**

Displays various environmental information to assist troubleshooting.

**System Log**

System log captures output from java.util.logging output related to Jenkins.

**Load Statistics**

Check your resource utilization and see if you need more computers for your builds.

**About Jenkins**

See the version and license information.

Jenkins

Search (⌘+K)

Dashboard > Manage Jenkins > Credentials

## Credentials

T	P	Store ↓	Domain	ID	Name
---	---	---------	--------	----	------

### Stores scoped to Jenkins

P	Store ↓	Domains
System		(global)

Icon: S M L

3

Jenkins

Search (⌘+K)

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)

## Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
This credential domain is empty. How about <a href="#">adding some credentials?</a>			

Icon: S M L

4

**Jenkins**

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### New credentials

Kind: **SSH Username with private key** (circled)

Scope: Global (Jenkins, nodes, items, all child items, etc)

ID: **ssh-prod\_instance** (circled)

Description: (empty)

Username: **tuchsanai** (circled)

Treat username as secret (circled)

**PRIVATE KEY** (Large red box around this section)

Enter directly:

Key:

```
-----BEGIN OPENSSH PRIVATE KEY-----
-----END OPENSSH PRIVATE KEY-----
```

Enter New Secret Below

Passphrase: (empty)

**Create**

**For SSH**

**Jenkins**

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
ssh-prod_instance	ssh-prod_instance	SSH Username with private key	

Icon: S M L (circled)

**Compute Engine**

VM instances

INSTANCES OBSERVABILITY INSTANCE SCHEDULES

VM instances

Filter: Enter primary name or value

Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
□	jenkins	asia-southeast1-a		10.148.0.15 (nic0)	34.126.119.107 (nic0)	SSH	⋮
□	prod-instance	asia-southeast1-b		10.148.0.16 (nic0)	34.143.151.212 (nic0)	SSH	⋮

Related actions:

- Explore Backup and DR (NEW)
- View billing report
- Monitor VMs
- Explore VM logs

## 2.Add Github-Credentials

Jenkins Dashboard

I will Learn Jenkins and get better paying job

Manage Jenkins

S	W	Name	Last Success	Last Failure	Last Duration
Green	Yellow	VeryFirstJenkinsJob	19 days #3	N/A	22 ms

Manage Jenkins

VeryFirstJenkinsJob

Build Queue

Build Executor Status

2

Manage Credentials

Configure Global Security

Manage Users

In-process Script Approval

Status Information

System Information

System Log

Load Statistics

3

Credentials

Stores scoped to Jenkins

P	Store	Domain	ID	Name
P	Store	Domains		
		(global)		

Add Credentials

4

Global credentials (unrestricted)

+ Add Credentials

This credential domain is empty. How about adding some credentials?

# For Github

New credentials

1. **Id**: Username with password

2. **Username**: tuchsanai

3. **Password**:

4. **ID**: github

Description: my github for test jenkins

Settings / Developer settings

Personal access tokens (classic)

Make sure to copy your personal access token now. You won't be able to see it again!

ghp\_Zp7azz7td4qxUcZdLyTT8PHG1Ljri0t3Fz

Personal access tokens (classic) | Jenkins | OAuth access tokens

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

repo

- repo:status
- repo\_deployment
- public\_repo
- repo:invite
- security\_events

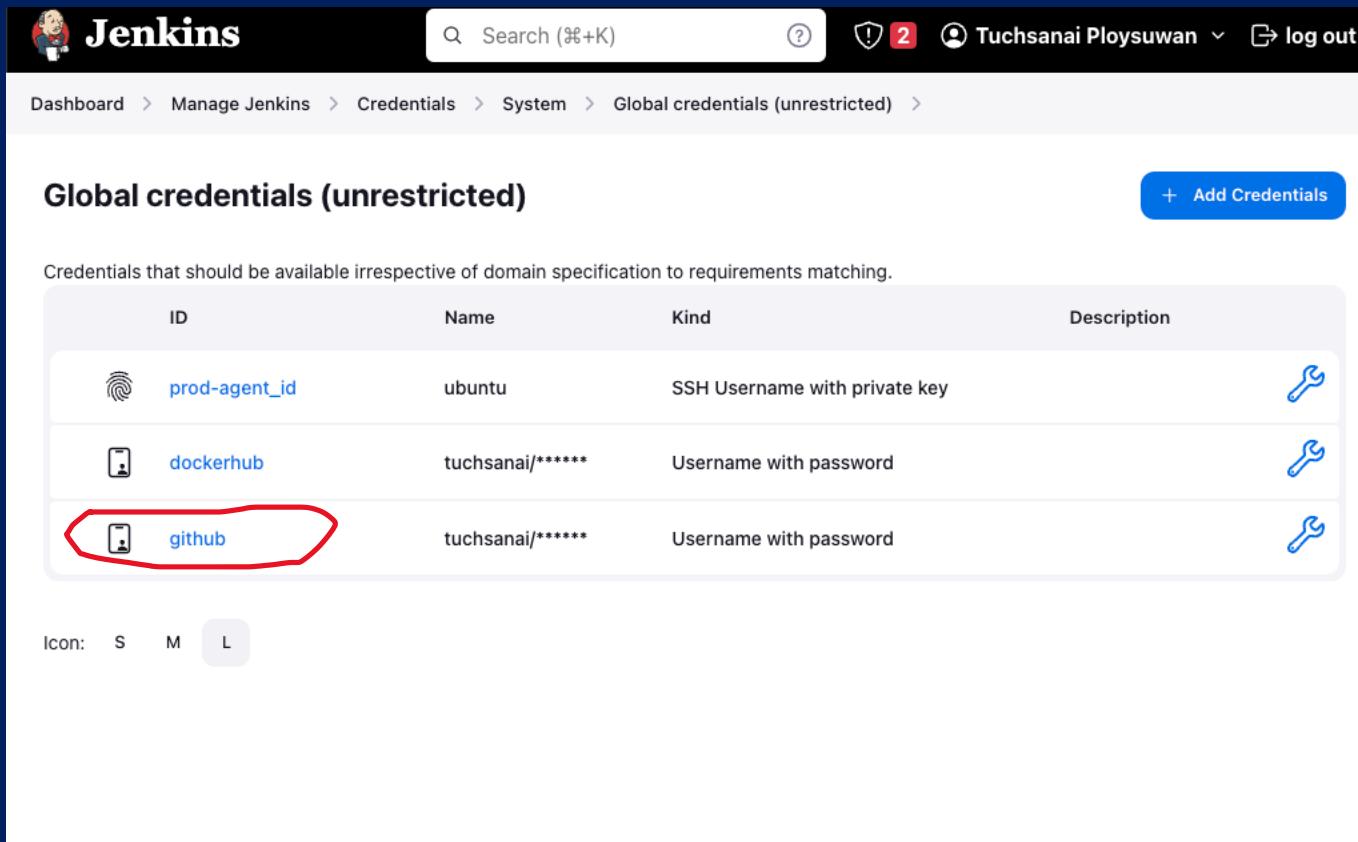
Full control of private repositories  
Access commit status  
Access deployment status  
Access public repositories  
Access repository invitations  
Read and write security events

admin:repo\_hook

- write:repo\_hook
- read:repo\_hook

Full control of repository hooks  
Write repository hooks  
Read repository hooks

# For Github



The screenshot shows the Jenkins Global credentials (unrestricted) page. The navigation path is: Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted). The page title is "Global credentials (unrestricted)". A blue button on the right says "+ Add Credentials". Below the title, a sub-instruction reads: "Credentials that should be available irrespective of domain specification to requirements matching." A table lists three credentials:

ID	Name	Kind	Description
	prod-agent_id	ubuntu	SSH Username with private key
	dockerhub	tuchsanai/********	Username with password
	github	tuchsanai/********	Username with password

At the bottom left, there are icons for S, M, and L, with L being highlighted.

### 3.Add docker hub Credentials

Jenkins Dashboard

I will Learn Jenkins and get better paying job

Manage Jenkins

S	W	Name	Last Success	Last Failure	Last Duration
Green	Yellow	VeryFirstJenkinsJob	19 days #3	N/A	22 ms

Manage Jenkins

VeryFirstJenkinsJob

Build Queue

Build Executor Status

Dashboard

Build Queue

Build Executor Status

Manage Nodes and Clouds

Manage Credentials

Configure Global Security

Manage Users

In-process Script Approval

Security

Status Information

System Information

System Log

Load Statistics

Configure Credential Providers

2

Jenkins

Dashboard > Manage Jenkins > Credentials

Credentials

T	P	Store	Domain	ID	Name
System	(global)				

Stores scoped to Jenkins

Domains

(global)

Icon: S M L

3

Jenkins

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Global credentials (unrestricted)

+ Add Credentials

ID	Name	Kind	Description
			This credential domain is empty. How about adding some credentials?

4

The screenshot shows the Docker Hub homepage. A large blue banner on the left features the text "Build and Ship any Application Anywhere". Below the banner, it says "Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications." A red hand-drawn number "5" is written over the blue banner area. At the bottom, it says "Docker Hub is the world's largest library and community for container images". The main navigation bar at the top includes "Explore", "Pricing", "Sign In", and "Sign up". A search bar says "Search Docker Hub".

Welcome back  
Sign in to Docker

Username or email address

Sign in

Don't have an account? Sign up

The screenshot shows the Docker Hub user profile page for the user "tuchsanai". The top navigation bar includes "Explore", "Repositories", "Organizations", "Search Docker Hub", and a user icon with a red hand-drawn letter "T". A red hand-drawn letter "b" is written near the bottom left of the page. The sidebar on the right has links for "What's New", "My Profile", "My Account" (which is circled in red), and "Billing". It also shows a "Sign out" button. The main content area lists three repositories: "tuchsanai / pytorch\_jupyterlab\_ubuntu22.04", "tuchsanai / kubirstapp", and "tuchsanai / custom-nginx-registry".

tuchsanai / pytorch\_jupyterlab\_ubuntu22.04  
Contains: Image | Last pushed: 2 months ago

tuchsanai / kubirstapp  
Contains: Image | Last pushed: 3 months ago

tuchsanai / custom-nginx-registry  
Contains: Image | Last pushed: 9 months ago

What's New

My Profile

**My Account**

Billing

Sign out

Community All-Hands: On-Demand

All sessions from our 6th Community All-Hands are now available on-demand! Over 35 talks cover best practices

The screenshot shows the Docker Hub account settings under the 'Security' tab. The user profile 'tuchsanai' is displayed, with a red circle highlighting it. A large red '7' is drawn next to the profile. Below the profile, the 'Access Tokens' section is shown, with a red circle highlighting the 'New Access Token' button. A large red '8' is drawn next to this button. The 'Two-Factor Authentication' section is also visible.

The screenshot shows the 'Copy Access Token' dialog box. It contains the generated token 'dckr\_pat\_tgAISGZbNsN5LuR4ckSI6Zb6c8', a 'Copy' button, and a warning message: 'WARNING: This access token will only be displayed once. It will not be stored and cannot be retrieved. Please be sure to save it now.' A large red '9' is drawn next to the token field.

The screenshot shows the Docker Hub account settings under the 'Security' tab after creating an access token. The 'Access Tokens' section now lists the token 'mydockhub' with details: Source: MANUAL, Scope: Read, Write, Delete, Last Used: Never, Created: Jan 04, 2024, 20:48:55. A large red '9' is drawn next to the token row. The 'Two-Factor Authentication' section is also visible.

**Jenkins**

Search (⌘+K) ? ! 2 Tuchsanai Ploysawan log out

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### New credentials

Kind ?  
Username with password

Scope ?  
Global (Jenkins, nodes, items, all child items, etc)

Username ?  
tuchsanai

Treat username as secret ?

Password ?  
.....

ID ?  
dockerhub

Description ?

Create

**Jenkins**

Search (⌘+K) ? ! 2 Tuchsanai Ploysawan log out

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 prod-agent_id	ubuntu	SSH Username with private key	
 dockerhub	tuchsanai/*****	Username with password	

Icon: S M L



## Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description	
<a href="#">github</a>	tuchsanai/******** (my github for test jenkins)	Username with password	my github for test jenkins	
<a href="#">dockerhub</a>	tuchsanai/******** (my docker hub)	Username with password	my docker hub	

Icon: S M L

# **LAB 3 : First Jenkins**



Jenkins

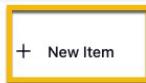
Search (⌘+K)



e



Dashboard >



1



Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

Built-In Node

(offline)

## Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job



Set up a distributed build

Set up an agent



Configure a cloud



Learn more about distributed builds





## Enter an item name

first\_project

» Required field

2

**Freestyle project**

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**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.

3

**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 separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**

Creates a set of multibranch project subfolders by scanning for repositories.

## Configure

### General

### Advanced Project Options

### Pipeline

 Throttle builds ?

#### Build Triggers

- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?
- Quiet period ?
- Trigger builds remotely (e.g., from scripts) ?

#### Advanced Project Options

Advanced ▾

### Pipeline

#### Definition

Pipeline script

#### Script ?

```
1~ pipeline {  
2~   agent any // Execute on any available Jenkins agent  
3~  
4~   stages {  
5~     stage('Hello World') {  
6~       steps {  
7~         echo 'Hello World!'  
8~       }  
9~     }  
10~   }  
11~ }  
12~  
13~
```

4

try sample Pipeline... ▾

 Use Groovy Sandbox ?

[https://github.com/Tuchsanai/DevTools/tree/main/03\\_Jenkins/03\\_First\\_Jenkins](https://github.com/Tuchsanai/DevTools/tree/main/03_Jenkins/03_First_Jenkins)

Name	Last commit message	Last commit date
..		
readme.md	ss	28 minutes ago

readme.md

First Jenkins Pipeline

```
pipeline {  
  agent any // Execute on any available Jenkins agent  
  
  stages {  
    stage('Hello World') {  
      steps {  
        sh echo 'Hello World!'  
      }  
    }  
  }  
}
```

Dashboard &gt; first\_project &gt;

 Status

&lt;/&gt; Changes

 Build Now

5

 Configure Delete Pipeline Full Stage View Rename Pipeline Syntax

## first\_project

 Add description Disable Project

### Stage View

No data available. This Pipeline has not yet run.

### Permalinks

 Build Historytrend ▾

No builds

 Atom feed for all  Atom feed for failures

 Status

## first\_project

&lt;/&gt; Changes

 Add description

▷ Build Now

 Disable Project Configure Delete Pipeline Full Stage View Rename Pipeline Syntax Build History

trend ▾

 Filter...

/

 #1

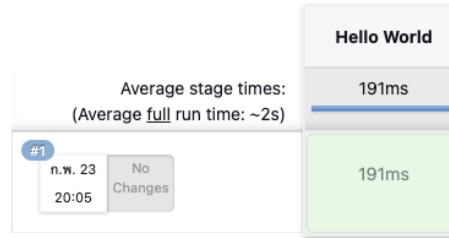
23 n.w. 2024 13:05

### Permalinks

6

### Stage View

Average stage times:  
(Average full run time: ~2s)



 Status

7

&lt;/&gt; Changes

 Console Output View as plain text Edit Build Information Delete build '#1' Restart from Stage Replay Pipeline Steps Workspaces

## Console Output

```
Started by user Tuchsanai Ploysawan
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/first_project
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello World)
[Pipeline] echo
Hello World!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

# LAB 4 : Jenkins Pipeline and Environment

[Click: Go to LAB](#)

## Declarative Pipeline fundamentals

In Declarative Pipeline syntax, the `pipeline` block defines all the work done throughout your entire Pipeline.

```
Jenkinsfile (Declarative Pipeline)
pipeline {
    agent any ①
    stages {
        stage('Build') { ②
            steps {
                // ③
            }
        }
        stage('Test') { ④
            steps {
                // ⑤
            }
        }
        stage('Deploy') { ⑥
            steps {
                // ⑦
            }
        }
    }
}
```

- ① Execute this Pipeline or any of its stages, on any available agent.
- ② Defines the "Build" stage.
- ③ Perform some steps related to the "Build" stage.
- ④ Defines the "Test" stage.
- ⑤ Perform some steps related to the "Test" stage.
- ⑥ Defines the "Deploy" stage.
- ⑦ Perform some steps related to the "Deploy" stage.

## Jenkins pipeline with write in Groovy language

<https://www.jenkins.io/doc/book/pipeline/>

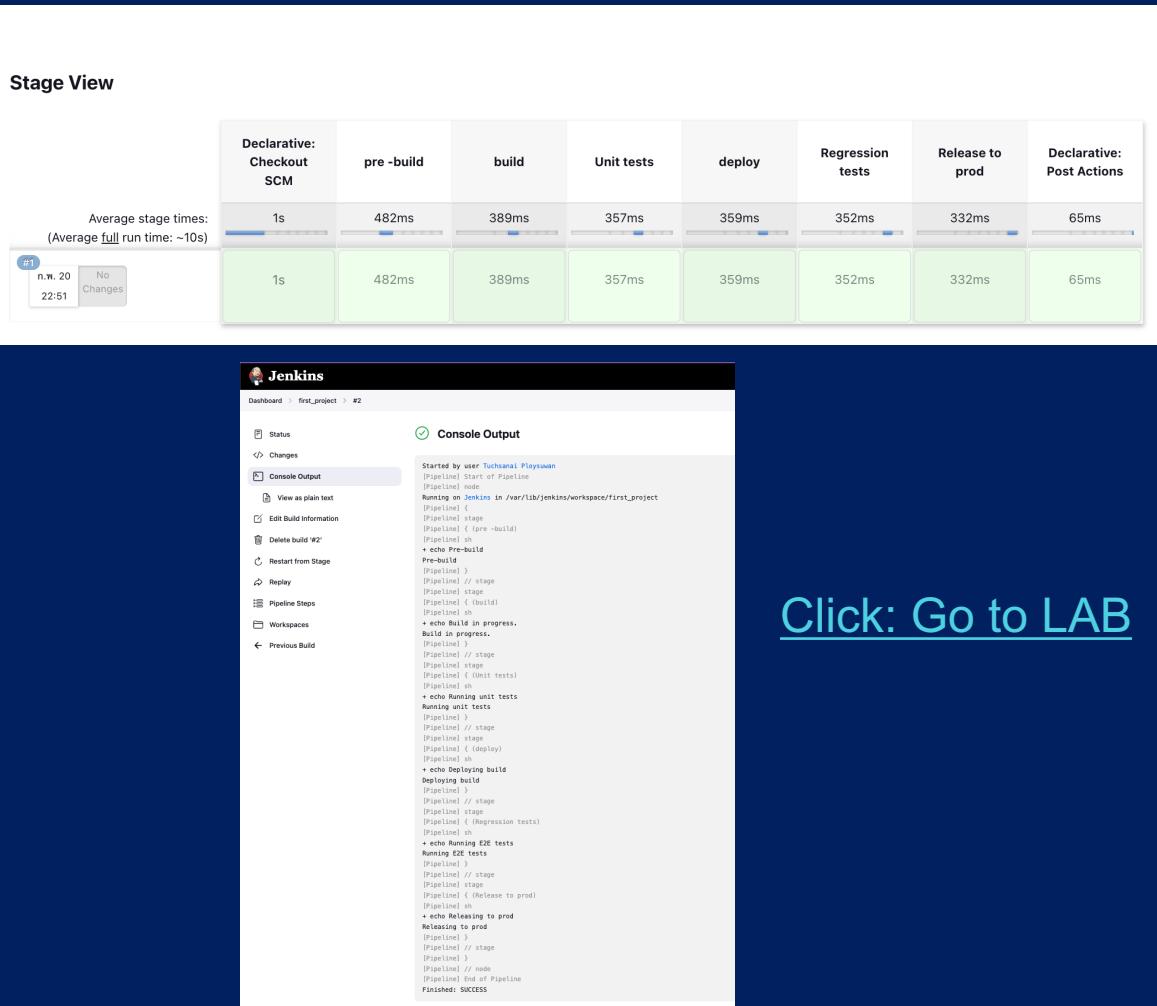
<https://www.jenkins.io/doc/pipeline/steps/workflow-cps/#pipeline-groovy>

# LAB 4.1: Pipeline with Multiple Stages

---

## Example 1. Multiple Stages

```
pipeline {  
    agent any  
    stages {  
        stage('pre-build') {  
            steps {  
                sh 'echo Pre-build'  
            }  
        }  
        stage('build') {  
            steps {  
                sh 'echo Build in progress.'  
            }  
        }  
        stage('Unit tests') {  
            steps {  
                sh 'echo Running unit tests'  
            }  
        }  
        stage('deploy') {  
            steps {  
                sh 'echo Deploying build'  
            }  
        }  
        stage('Regression tests') {  
            steps {  
                sh 'echo Running E2E tests'  
            }  
        }  
        stage('Release to prod') {  
            steps {  
                sh 'echo Releasing to prod'  
            }  
        }  
    }  
}
```



# LAB 4.2: Pipeline with post actions

---

# Pipeline with post actions

- Run additional steps at the end of pipeline or stage
- Not a required field
- Used to handle failure conditions
- Supports different conditions

## Example 2. Post actions and Conditions

```
pipeline {
    agent any
    stages {
        stage('build') {
            steps {
                sh 'python --version'
            }
        }
    }
    post {
        always {
            echo 'Always'
        }
        success {
            echo 'Only on SUCCESS'
        }
        failure {
            echo 'Only on Failure'
        }
        unstable {
            echo 'Only if run is unstable'
        }
        changed {
            echo 'Only if status changed from Success to Failure or vice versa w.r.t. last run.'
        }
    }
}
```

[Click: Go to LAB](#)

# If there are not python in your system

 Jenkins

Dashboard > post\_action >

Status post\_action

</> Changes  
▷ Build Now  
⚙ Configure  
Delete Pipeline  
Full Stage View  
Rename  
Pipeline Syntax

Build History trend ▾  
#1 26 n.w. 2024 08:47  
Atom feed for all Atom feed for failures

Stage View

Average stage times:

build	Declarative: Post Actions
484ms	144ms

#1 n.w. 26 15:47 No Changes 484ms failed

Permalinks

- Last build (#1), 6 min 34 sec ago
- Last failed build (#1), 6 min 34 sec ago
- Last unsuccessful build (#1), 6 min 34 sec ago
- Last completed build (#1), 6 min 34 sec ago

## Console Output

Started by user Tuchsanai Ploysuwan  
[Pipeline] Start of Pipeline  
[Pipeline] node  
Running on Jenkins in /var/lib/jenkins/workspace/post\_action  
[Pipeline] {  
[Pipeline] stage  
[Pipeline] { (build)  
[Pipeline] sh  
+ python --version  
/var/lib/jenkins/workspace/post\_action@tmp/durable-82f3b965/script.sh: line 1: python: command not found  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] stage  
[Pipeline] { (Declarative: Post Actions)  
[Pipeline] echo  
Always  
[Pipeline] echo  
Only on Failure  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] }  
[Pipeline] // node  
[Pipeline] End of Pipeline  
ERROR: script returned exit code 127  
Finished: FAILURE

If there are python in your system

# Jenkins

Dashboard > post\_action >

**post\_action**

Status Changes Build Now Configure Delete Pipeline Full Stage View Rename Pipeline Syntax

Build History trend Filter... #3 n.w. 26 16:07 No Changes

Average stage times: (Average full run time: ~977ms)

build	Declarative: Post Actions
354ms	121ms

Permalinks Atom feed for all Atom feed for failures

## Console Output

```
Started by user Tuchsanai Ploysuwan
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/post_action
[Pipeline] {
[Pipeline] stage
[Pipeline] { (build)
[Pipeline] sh
+ python3 --version
Python 3.10.12
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Always
[Pipeline] echo
Only if status changed from Success to Failure or vice versa w.r.t. last run.
[Pipeline] echo
Only on SUCCESS
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

**always**

Run the steps in the `post` section regardless of the completion status of the Pipeline's or stage's run.

**changed**

Only run the steps in `post` if the current Pipeline's or stage's run has a different completion status from its previous run.

**fixed**

Only run the steps in `post` if the current Pipeline's or stage's run is successful and the previous run failed or was unstable.

**regression**

Only run the steps in `post` if the current Pipeline's or stage's run's status is failure, unstable, or aborted and the previous run was successful.

**aborted**

Only run the steps in `post` if the current Pipeline's or stage's run has an "aborted" status, usually due to the Pipeline being manually aborted. This is typically denoted by gray in the web UI.

**failure**

Only run the steps in `post` if the current Pipeline's or stage's run has a "failed" status, typically denoted by red in the web UI.

**success**

Only run the steps in `post` if the current Pipeline's or stage's run has a "success" status, typically denoted by blue or green in the web UI.

**unstable**

Only run the steps in `post` if the current Pipeline's or stage's run has an "unstable" status, usually caused by test failures, code violations, etc. This is typically denoted by yellow in the web UI.

**unsuccessful**

Only run the steps in `post` if the current Pipeline's or stage's run has not a "success" status. This is typically denoted in the web UI depending on the status previously mentioned.

**cleanup**

Run the steps in this `post` condition after every other `post` condition has been evaluated, regardless of the Pipeline or stage's status.

# LAB 4.3: Pipeline Environment Variables

---

## - Basic environment variables 1

```
pipeline {  
    agent any  
  
    environment {  
        // Define environment variables  
        MY_ENV_VAR = 'Hello, Jenkins Environment Variables!'  
        ANOTHER_VAR = 'This is another environment variable.'  
    }  
  
    stages {  
        stage('Demo') {  
            steps {  
  
                // Use the environment variables  
                echo "Using environment variable: ${env.MY_ENV_VAR}"  
                echo "Using another environment variable: ${env.ANOTHER_VAR}"  
  
                // Set a new environment variable or modify an existing one  
                script {  
                    env.NEW_VAR = 'This is a new environment variable set during runtime.'  
                }  
  
                echo "Using a newly set environment variable: ${env.NEW_VAR}"  
                sh 'printenv'  
            }  
        }  
    }  
}
```

[Click: Go to LAB](#)

## Console Output

```
Started by user Tuchsanai Ploysawan
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/env_variable
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Demo)
[Pipeline] echo
Using environment variable: Hello, Jenkins Environment Variables!
[Pipeline] echo
Using another environment variable: This is another environment variable.
[Pipeline] script
[Pipeline] {
[Pipeline] }
[Pipeline] // script
[Pipeline] echo
Using a newly set environment variable: This is a new environment variable set during runtime.
[Pipeline] sh
+ printenv
JENKINS_HOME=/var/lib/jenkins
USER=jenkins
CI=true
RUN_CHANGES_DISPLAY_URL=http://54.251.180.144:8080/job/env_variable/4/display/redirect?page=changes
NODE_LABELS=builder
HUDSON_URL=http://54.251.180.144:8080/
HOME=/var/lib/jenkins
BUILD_URL=http://54.251.180.144:8080/job/env_variable/4/
HUDSON_COOKIE=50d258a1-bba2-4618-b4f0-beac4dc27368
JENKINS_SERVER_COOKIE=durable-bb56faeff47a3330ae9e322e09b2ce9ed8d56fa52e54949893ce25db758a8fe5
NOTIFY_SOCKET=/run/systemd/notify
SYSTEM_EXECPID=8226
WORKSPACE=/var/lib/jenkins/workspace/env_variable
LOGNAME=jenkins
NODE_NAME=jenkins
NODE_NAME=jenkins
JOURNAL_STREAM=8:46337
NEW_VAR=This is a new environment variable set during runtime.
RUN_ARTIFACTS=5139CATC_0NE_HStP772Yz91z0p9144180880/job/env_variable/4/display/redirect?page=artifacts
STAGE_NAME=Dev
EXECUTOR_NUMBER=0
RUN_TESTS_DISPLAY_URL=http://54.251.180.144:8080/job/env_variable/4/display/redirect?page=tests
BUILD_DISPLAY_NAME=#4
HUDSON_HOME=/var/lib/jenkins
JOB_BASE_NAME=env_variable
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin
INVOCATION_ID=3d106676f2440d98512@e5fa349ed
BUILD_ID=4
BUILD_TAGjenkins-env_variable-4
LANG=C.UTF-8
JENKINS_URL=http://54.251.180.144:8080/
JOB_URL=http://54.251.180.144:8080/job/env_variable
ANOTHER_VAR=This is another environment variable.
MY_ENV_VAR>Hello, Jenkins Environment Variables!
```

## - Basic environment variables 2

```
Jenkinsfile (Declarative Pipeline)
pipeline {
    agent any

    environment {
        DISABLE_AUTH = 'true'
        DB_ENGINE    = 'sqlite'
    }

    stages {
        stage('Build') {
            steps {
                echo "Database engine is ${DB_ENGINE}"
                echo "DISABLE_AUTH is ${DISABLE_AUTH}"
                sh 'printenv'
            }
        }
    }
}
```



## Console Output

```
Started by user Tuchsanai Ploysuwan
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/env_variable
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] }
[Pipeline] echo
Database engine is sqlite
[Pipeline] echo
DISABLE_AUTH is true
[Pipeline] sh
+ printenv
DISABLE_AUTH=true
JENKINS_HOME=/var/lib/jenkins
DB_ENGINE=sqlite
USER=jenkins
CI=true
RUN_CHANGES_DISPLAY_URL=http://54.251.180.144:8080/job/env_variable/7/display/redirect?page=changes
NODE_LABELS=built-in
HUDSON_URL=http://54.251.180.144:8080/
HOME=/var/lib/jenkins
```

# LAB 4.4: Docker and Environment variables

---

## - Docker and Environment variables

```
pipeline {  
    agent any // Execute on any available Jenkins agent  
  
    environment {  
        // Define an environment variable  
        DOCKER_VERSION = ''  
    }  
  
    stages {  
        stage('Check Docker') {  
            steps {  
                script {  
                    // Try to get Docker version  
                    def dockerCheck = sh(script: 'docker --version || echo "No Docker"', returnStdout: true).trim()  
  
                    sh 'echo dockerCheck = ' + dockerCheck  
  
                    // Check if Docker is available and set environment variable accordingly  
                    if (dockerCheck.contains('Docker version')) {  
                        env.DOCKER_VERSION = dockerCheck  
                    } else {  
                        env.DOCKER_VERSION = 'No Docker'  
                    }  
                }  
            }  
        }  
  
        stage('Run Docker Hello World') {  
            steps {  
                script {  
                    // Check if Docker version was found and run hello-world image  
                    if (env.DOCKER_VERSION != 'No Docker') {  
                        sh 'docker run hello-world'  
                    } else {  
                        echo 'No Docker available on this machine'  
                    }  
                }  
            }  
        }  
    }  
}
```

[Click: Go to LAB](#)

# Jenkins

Search (⌘+K)

Dashboard > docker >

Status

</> Changes

▷ Build Now

⚙ Configure

>Delete Pipeline

🔍 Full Stage View

✍ Rename

ⓘ Pipeline Syntax

Build History trend

Filter... /

#1 26 n.w. 2024 09:41

Atom feed for all Atom feed for failures

## docker

### Stage View

	Check Docker	Run Docker Hello World
Average stage times:	695ms	647ms
(Average full run time: ~1s)		
n.w. 26 16:41	No Changes	
	695ms	647ms

### Permalinks

## Console Output

```
Started by user Tuchsanai Ploysawan
[Pipeline] Start of Pipeline
[Pipeline] node
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ docker --version
[Pipeline] sh
+ echo dockerCheck = Docker version 25.0.3, build 4debf41
dockerCheck = Docker version 25.0.3, build 4debf41
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] sh
+ docker run hello-world
[Pipeline] }

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

[Pipeline] }

[Pipeline] // script

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

[Pipeline] // withEnv

[Pipeline] }

[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS

