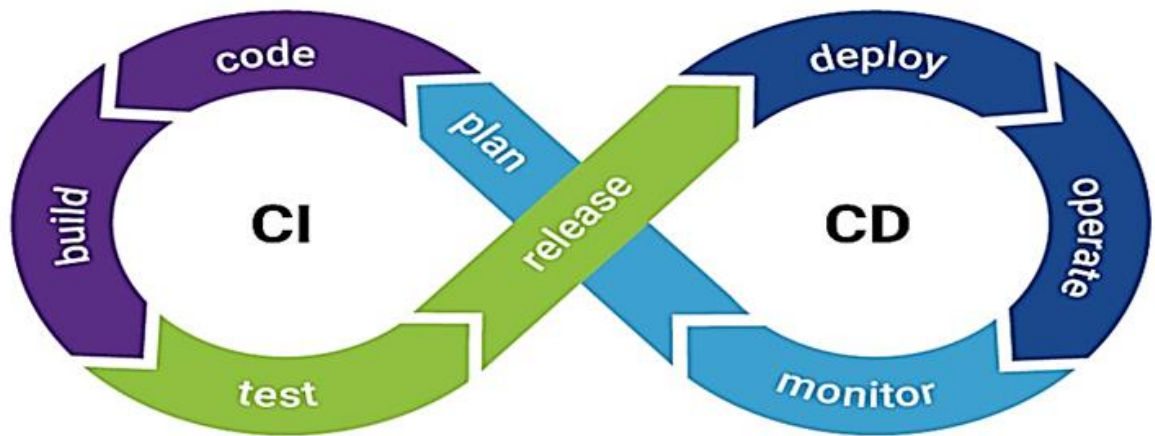


As part of this article let's have deep dive on Continuous Integration and Continuous Delivery with AnyPoint Platform - CloudHub.

Before we begin to understand CICD in CloudHub let's first touch base on few jargons being use in the article.

- **CloudHub** is an integration platform as a service (iPaaS) where you can deploy cross-cloud integration applications in the cloud, create new APIs on top of existing data sources, integrate on-premises applications with cloud services, and much more.
- **Continuous Integration** An approach to be continually validating the state of a codebase through automated testing. Achieved through integration with source control/version control and artifact store.
- **Continuous Delivery / Deployment** An approach to regularly deploying artifacts that successfully pass the CI phase to ensure confidence around the deployment
- **API Auto Discovery** API Autodiscovery is a mechanism that manages an API from API Manager by pairing the deployed application to an API created on the platform. API Management includes tracking, enforcing policies if you apply any, and reporting API analytics. Critical to the Autodiscovery process is identifying the API by providing the API name and version.
- **API Policies** Policies enable you to enforce regulations to help manage security, control traffic, and improve adaptability of your APIs. For example, a policy can control authentication, access, allotted consumption, and service level access (SLA). You can implement all these regulations with no modification to the code implementation. Mulesoft provides ready-to-use default policies that are shipped with the product.



Pre requisites :

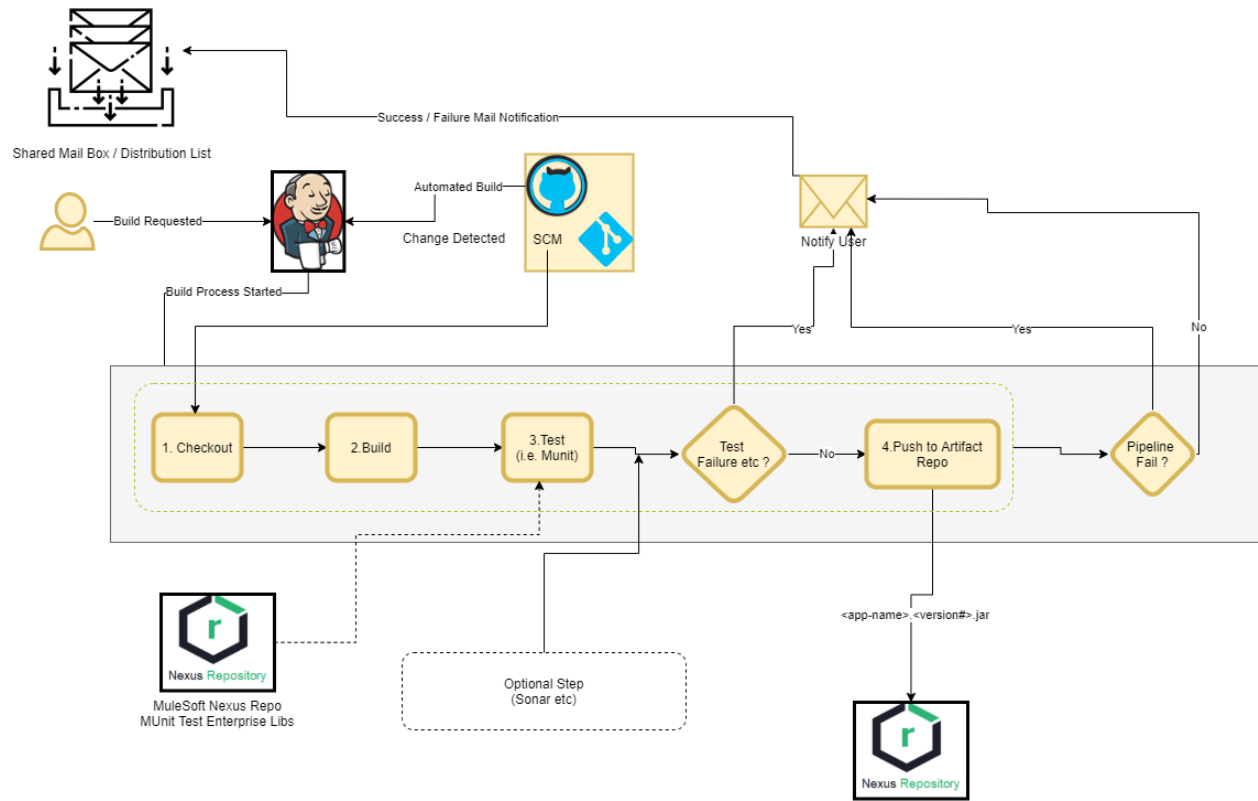
CICD is a culture set of best practices to be followed to have code changes deployed more frequently and reliably.

Before understanding the overall CICD with AnyPoint Platform - CloduHub we should have knowledge of how API Deployment , Auto Discovery and Policy works in CloudHub.

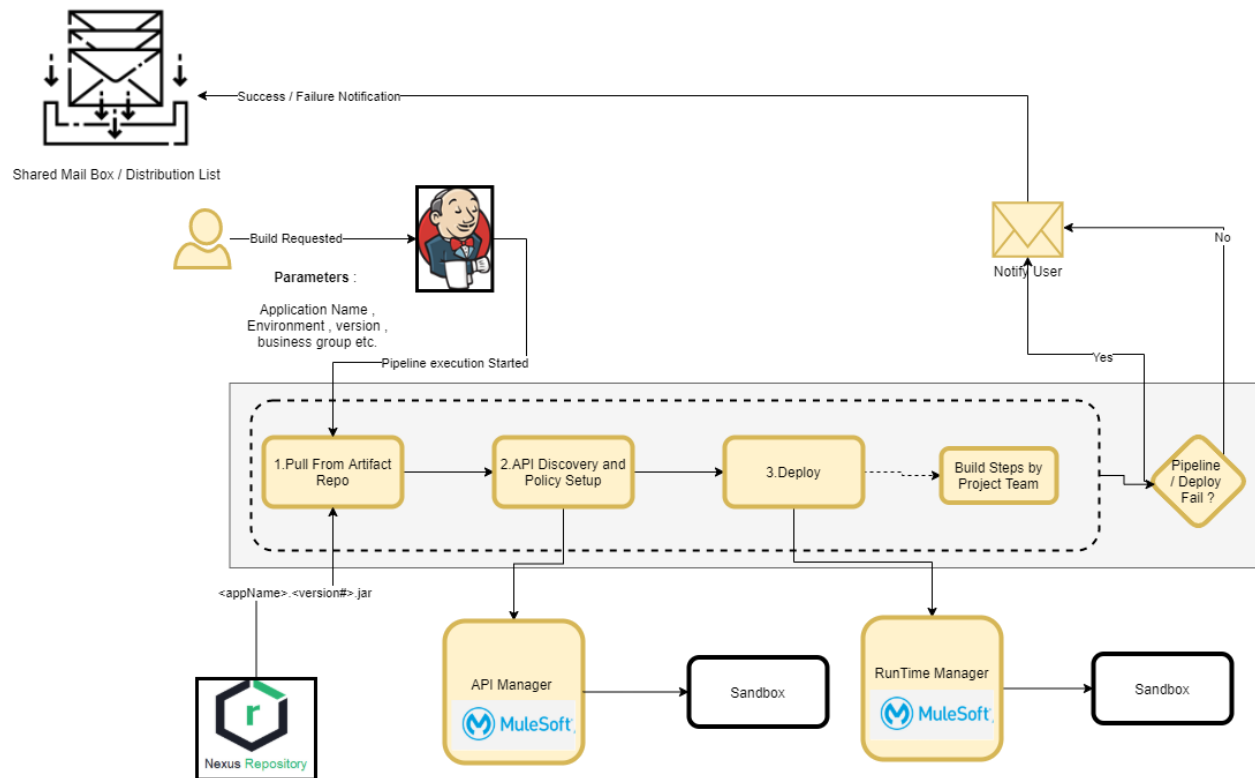
Refer these topics for more details on [API Deployment](#) , [Auto Discovery](#) and [Policies](#) works in CloudHub

Basic understanding of Anypoint platform

CI Flow



CD Flow



So let's begin the setup

Tools with Initial Instruction to setup the Environment :

- Maven [Download From Here](#)
- Git [Download From Here](#)
- Jenkins (make sure you have below plugins installed)
 - [CloudHub Deployer](#)

- [Maven Integration plugin](#)

Maven

Maven installations Add Maven

 Maven

Name

MAVEN_HOME

☐ Install automatically ?

Delete Maven

- [Groovy](#)

- [Nexus Artifact Uploader](#)


- [Email Extension Plugin](#)

- [Pipeline Utility Steps](#)

- [Git plugin](#)

Git

Git installations

 **Git**

Name

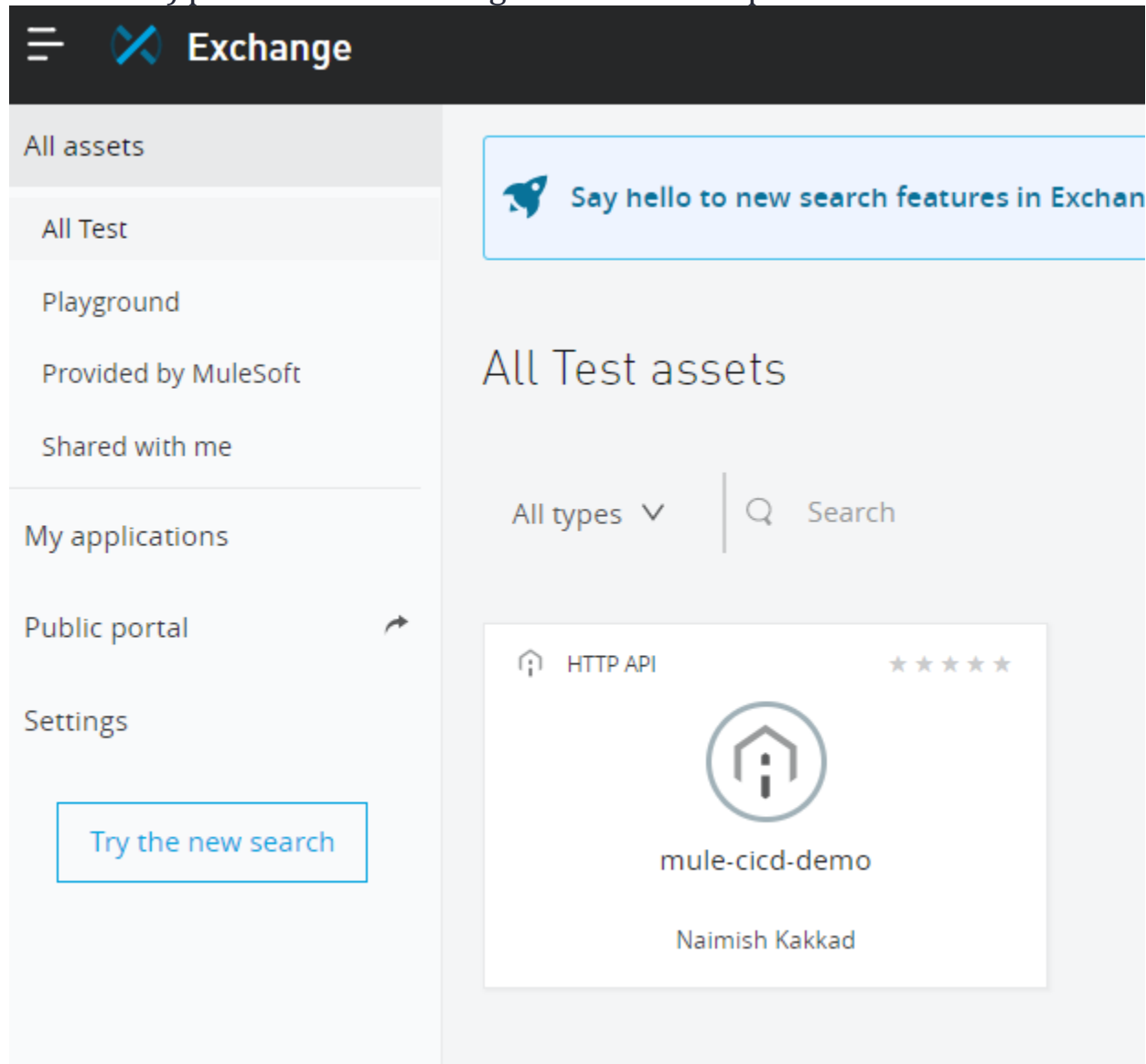
Path to Git executable

☐ Install automatically ?

Add Git ▼ Delete Git

- AnyPoint Studio [Download From Here](#)
- Nexus repo [Download from here](#)
- Active gmail account
- Active Anypoint Platform Account [Sign Up Here](#)

- Make sure you have specification/simple api asset (i.e. mule-cicd-demo) published in exchange like below snapshot



- Github [Sign Up Here](#)

Please follow below steps before starting with Jenkins Pipeline configurations

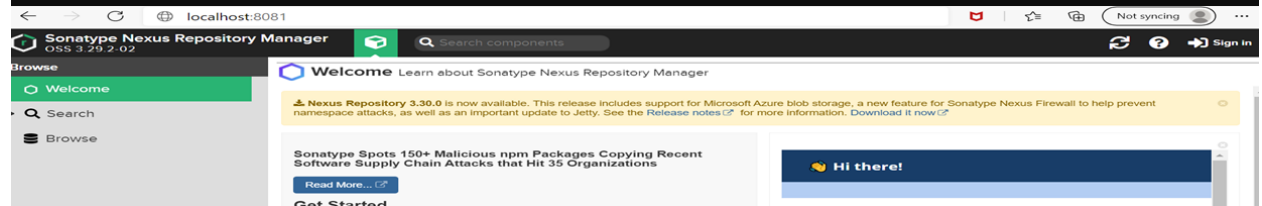
Nexus Repo OSS :

Verify Installation

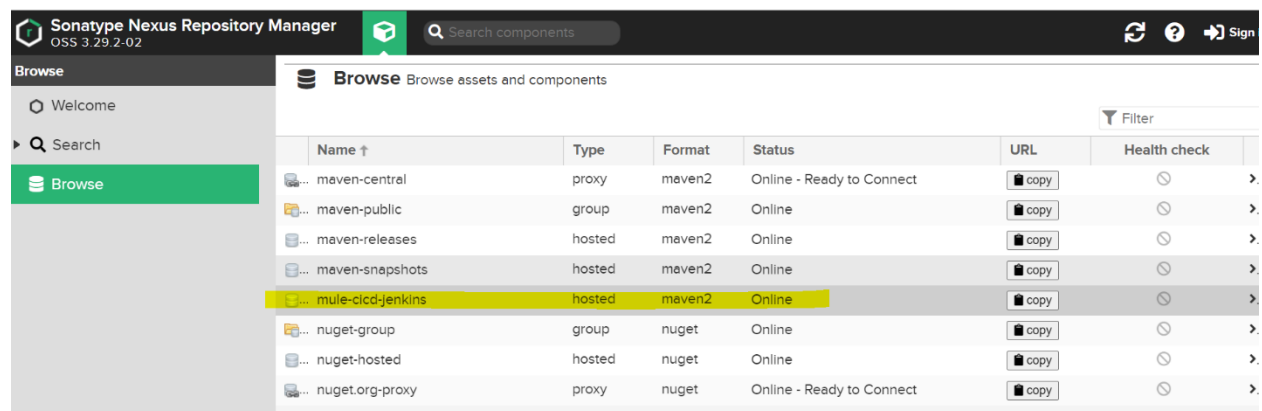
```
C:\Work\nexus-3.29.2-02-win64\nexus-3.29.2-02\bin>nexus /run
2021-03-23 14:13:41,438+0530 INFO [FelixStartLevel] *SYSTEM org.sonatype.nexus.pax.logging.NexusLogActivator - start
2021-03-23 14:13:41,954+0530 INFO [FelixStartLevel] *SYSTEM org.sonatype.nexus.features.internal.FeaturesWrapper - Fast FeaturesService starting
2021-03-23 14:13:43,154+0530 WARN [FelixStartLevel] *SYSTEM uk.org.lidalia.sysoutslf4j.context.SysOutOverSLF4JInitialiser - Your logging framework class org.ops4j.pax.logg
ing.slf4j.Slf4jLogger is not known
2021-03-23 14:13:43,154+0530 INFO [FelixStartLevel] *SYSTEM uk.org.lidalia.sysoutslf4j.context.SysOutOverSLF4J - Package org.ops4j.pax.logging.slf4j registered; all classe
s within it or subpackages of it will be allowed to print to System.out and System.err
2021-03-23 14:13:43,154+0530 INFO [FelixStartLevel] *SYSTEM uk.org.lidalia.sysoutslf4j.context.SysOutOverSLF4J - Replaced standard System.out and System.err PrintStreams w
ith SLF4JPrintStreams
2021-03-23 14:13:43,154+0530 INFO [FelixStartLevel] *SYSTEM uk.org.lidalia.sysoutslf4j.context.SysOutOverSLF4J - Redirected System.out and System.err to SLF4J for this con
text

default ValidatorFactory
2021-03-23 14:14:20,831+0530 INFO [jetty-main-1] *SYSTEM org.sonatype.nexus.siesta.SiestaServlet - Initialized
2021-03-23 14:14:20,837+0530 INFO [jetty-main-1] *SYSTEM org.sonatype.nexus.repository.httpbridge.internal.ViewServlet - Initialized
2021-03-23 14:14:21,355+0530 INFO [jetty-main-1] *SYSTEM org.eclipse.jetty.server.handler.ContextHandler - Started o.e.j.w.WebAppContext@333cba3c(Sonatype Nexus,/
C:/Work/nexus-3.29.2-02-win64/nexus-3.29.2-02/public/,AVAILABLE)
2021-03-23 14:14:21,408+0530 INFO [jetty-main-1] *SYSTEM org.eclipse.jetty.server.AbstractConnector - Started ServerConnector@406f1159(HTTP/1.1, (http/1.1)){0.0.0
2021-03-23 14:14:21,408+0530 INFO [jetty-main-1] *SYSTEM org.eclipse.jetty.server.Server - Started @45215ms
2021-03-23 14:14:21,416+0530 INFO [jetty-main-1] *SYSTEM org.sonatype.nexus.bootstrap.jetty.JettyServer -

-----
Started Sonatype Nexus OSS 3.29.2-02
-----
```



- Create maven2 hosted repo to save artifacts produced from CI Flow.



Name ↑	Type	Format	Status	URL	Health check
maven-central	proxy	maven2	Online - Ready to Connect	copy	health
maven-public	group	maven2	Online	copy	health
maven-releases	hosted	maven2	Online	copy	health
maven-snapshots	hosted	maven2	Online	copy	health
mule-cicd-jenkins	hosted	maven2	Online	copy	health
nuget-group	group	nuget	Online	copy	health
nuget-hosted	hosted	nuget	Online	copy	health
nuget.org-proxy	proxy	nuget	Online - Ready to Connect	copy	health

Jenkins :

Credentials Setup :

Credentials

T	P	Store	Domain	ID	Name
	Jenkins	(global)	git	naimishkakkad@gmail.com/*****	(github credentials)
	Jenkins	(global)	nexus-user-credentials	admin/*****	
	Jenkins	(global)	anypoint.username	anypoint.username	
	Jenkins	(global)	anypoint.password	anypoint.password	
	Jenkins	(global)	PLAYGROUND	Business Group	
	Jenkins	(global)	PLAYGROUND_SANDBOX	Environment	
	Jenkins	(global)	cloudhub.credentials	*****	(cloudhub.credentials)

Icon: S M L

Environment Information

Environment

Environment name: Sandbox
Environment ID: 23454f0c-507b-4d1b-aef
Client credentials
Client ID: 4e9c9eaf93fa40f7af16b2
Client secret:

Business Group

Business Group name: Playground
Business Group ID: 9c3b02a7-4aa5-4efb-bbt
Client credentials
Client ID: a8db658bee8c4f84a447
Client secret:

Create CI Pipeline :

(As I have already have existing job with same name it is giving me error but it will not occur for you as you are creating the same for the first time)

Enter an item name

» A job already exists with the name 'MuleSoft_CI_Nexus_Dzone'



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.



Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



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.

Select Pipeline Script :

Download Pipeline Script from here :

<https://github.com/naimishkakkad/jenkinspipelines/blob/main/CIPipeline.txt>

Sample Code - <https://github.com/naimishkakkad/mule-cicd-demo>

(Clone this and try this in your own GitHub repo)

localhost:8080/job/MuleSoft_CI_Nexus_Dzone/configure

MuleSoft_CI_Nexus_Dzone

General Build Triggers Advanced Project Options **Pipeline**

Pipeline

Definition Pipeline script

Script

```

1 pipeline {
2   agent any
3   tools {
4     maven "maven"
5   }
6   stages {
7     stage('Code Checkout') {
8       steps {
9         git branch: 'main',
10          credentialsId: 'git',
11          url: 'https://github.com/naimishkakkad/mule-cicd-demo.git'
12       }
13     }
14     stage('Build') {
15       steps {
16         bat "mvn -B -U -e -V clean -DskipTests package"
17       }
18     }
19   }
20 }

```

Use Groovy Sandbox

Pipeline Syntax

Point this to you Github repo

Jenkins

search

Dashboard MuleSoft_CI_Nexus_Dzone

Back to Dashboard

Status

Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Build History trend

find

Pipeline MuleSoft_CI_Nexus_Dzone

add description

Disable Project

Recent Changes

Stage View

Average stage times:
(Average full run time: ~28s)

Declarative: Tool Install	Code Checkout	Build	Test	Publish to Nexus Repository	Declarative: Post Actions
289ms	4s	11s	416ms	2s	4s
259ms	4s	9s	453ms	2s	5s

Mar 22 21:25 1 commit


Create CD Pipeline :

(As I have already have existing job with same name it is giving me error but it will not occur for you as you are creating the same for the first time)


Enter an item name

MuleSoft_CD_From_Nexus_Dzone


» A job already exists with the name 'MuleSoft_CD_From_Nexus_Dzone'

**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.

**Maven project**

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

**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.

Select Pipeline Script :

Download Pipeline Script from here :

<https://github.com/naimishkakkad/jenkinspipelines/blob/main/CDPipeline.txt>

Sample Code - <https://github.com/naimishkakkad/mule-cicd-demo>

(Clone this and try this in your own GitHub repo)

Add Parameters to the pipeline like below

☒ This project is parameterized

String Parameter

Name

project_name

Default Value

mule-cicd-demo

Description

Project name

Pipeline MuleSoft_CD_From_Nexus_Dzone

This build requires parameters:

project_name

mule-cicd-demo

version

1.0.1

business_group

PLAYGROUND

target_env

SANDBOX

cloudhub_app_name

mule-cicd-demo-jenkins

env_client_id

Concealed

Change Password

env_client_secret

Concealed

Change Password

Build

Pipeline

Definition

Pipeline script

Script

```
10 import org.jenkinsci.plugins.cloudhubdeployer.data.*
11 pipeline {
12     agent any
13     tools {
14         maven "maven"
15     }
16     options {
17         skipDefaultCheckout(true)
18     }
19     stages {
20         stage('Fetch Configuration Data') {
21             steps {
22                 cleanWs()
23                 git branch: 'main',
24                     credentialsId: 'git',
25                     url: 'https://github.com/naimishkakkad/mule-cicd-demo.git'
26             }
27         }
28     }
29 }
```

☐ Use Groovy Sandbox

Jenkins

search

Naimish Kakkad

log out

Dashboard

MuleSoft_CD_From_Nexus_Dzone

Back to Dashboard

Status

Changes

Build with Parameters

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Build History

find

Pipeline MuleSoft_CD_From_Nexus_Dzone

add description

Disable Project

Recent Changes

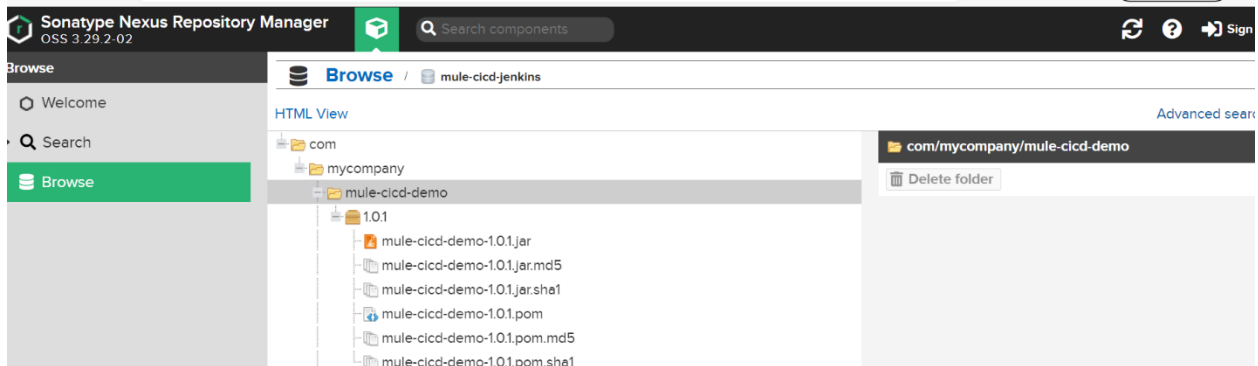
Stage View

Average stage times:
(Average full run time: ~2min 24s)

	Declarative: Tool Install	Fetch Configuration Data	Fetch Artifact	API Discovery and Policy Setup	Deploy	Declarative: Post Actions
#6	221ms	4s	9s	5s	5min 5s	3s
Mar 22 22:17	188ms	5s	7s	7s	1min 55s	6s

That's it :)

Just Hit the "Build Now" button for CI Pipeline and verify if Jar is published to Nexus repo or not.



Just Hit the "Build with parameters" button for CD Pipeline and verify if application is deployed to CloudHud or not.

Verify API Instance in API Manager

API Name	Version	Status	Client Applications	Creation Date
▼ mule-cicd-demo				1 version
	v1	● Active	0	03-22-2021 21:33

mule-cicd-demo v1

API Status: ● Active Asset Version: 1.0.0 Latest Type: HTTP

[Add consumer endpoint](#)

Mule runtime version: 4.3.0-20210319

Actions ▾

[View API in Exchange](#) >

[View configuration details](#) >

[View Analytics Dashboard](#) >

Automated Policies

There are no automated policies applied for the selected runtime version.

API level policies

[Apply New Policy](#)

[Edit policy order](#)

Name	Category	Fulfills	Requires
> Client ID enforcement ⓘ	Compliance	Client ID required	API Specification snippet
> JSON threat protection ⓘ	Security	JSON threat protected	

Magic of triggering API Policy setup and API Discovery dynamically via script is happening from [api.platform.properties](#)

Current support of these script is with JSON Threat protection , client id enforcement and JWT.

As part of this article we are using Client ID Enforcement and JSON Threat Protection.

Verify API Deployment in Runtime Manager

Runtime Manager

Playground

?

NK

SANDBOX

Applications

Servers

Alerts

VPCs

Load Balancers

Deploy application

Search Applications

All Applications (2)

Update Available (0)

Name ^	Server	Status	Runtime Version	Date Modified
mule-cicd-demo	CloudHub	Undeployed	4.3.0	2021-03-22 14:50:31
mule-cicd-demo-jenkins	CloudHub	Started	4.3.0	2021-03-23 18:13:06

[Switch back to classic applications list](#)

Happy Deploying the API to your Anypoint Platform using these CICD Pipelines along with API Discovery and Policy Setup.

Please find presentation here - [Detailed Presentation](#)