Lab (2)

1) What is Jenkins pipeline?

Jenkins Pipeline is a suite of plugins that 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.

2) What scripting language is Jenkins pipeline syntax based on?

Jenkins pipeline syntax is based on Groovy scripting language

3) What are the different ways to trigger pipeline?

There are many different ways to trigger a pipeline. Some of the most common ways include:

Manually triggering a pipeline: This is the simplest way to trigger a pipeline. You can do this by clicking the "Run" button in the Jenkins UI, or by using the jenkins.model.Jenkins.instance.run() method in the Groovy API.

Triggering a pipeline on a schedule: This allows you to run a pipeline at a specific time or interval. You can do this by configuring a schedule trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.schedule() method in the Groovy API.

Triggering a pipeline on a change in a repository: This allows you to run a pipeline whenever a change is made to a repository. You can do this by configuring a git trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.github() method in the Groovy API.

Triggering a pipeline on an event: This allows you to run a pipeline in response to an event, such as a new issue being created or a build failing. You can do this by configuring an event trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.buildTrigger() method in the Groovy API.

Triggering a pipeline from another pipeline: This allows you to run a pipeline as a dependency of another pipeline. You can do this by configuring a pipeline resource trigger in the Jenkins UI, or by using the jenkins.triggers.Trigger.pipelineResourceTrigger() method in the Groovy API.

4) What is different between parameter and jenkins env variable?

Parameters and environment variables are both used to pass values to Jenkins pipelines. However, they have different purposes and are used in different ways.

Parameters are values that are passed to a pipeline when it is run. They are typically used to configure the pipeline, such as the branch to build or the environment to use. Parameters can be set in the Jenkins UI, in a build script, or in a pipeline.

Environment variables are values that are available to all stages of a pipeline. They can be used to store configuration information, such as the database password or the API key. Environment variables can be set in the Jenkins UI, in a build script, or in a pipeline.

5) What is organization folder job and what is used for?

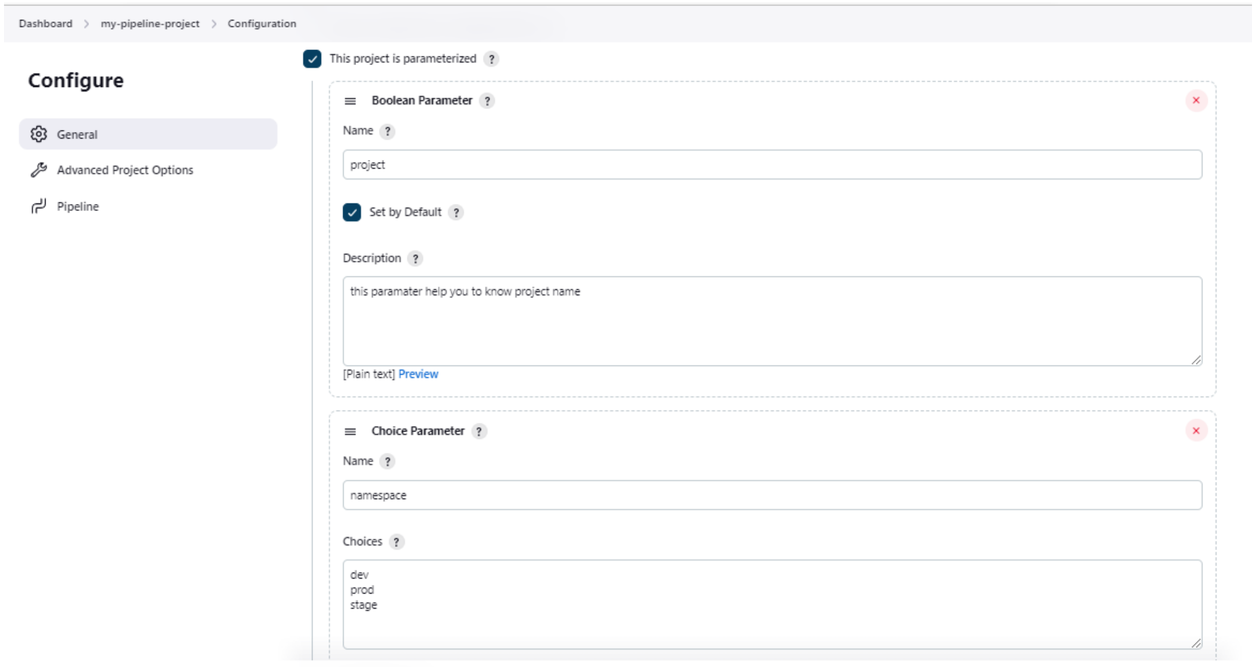
An Organization Folder Job is a type of job that allows you to automatically create and manage Multibranch Pipeline jobs for all repositories in an organization. This can be useful for large organizations with many repositories, as it can help to automate the process of creating and managing Jenkins jobs.

Once the Organization Folder Job has been created, Jenkins will automatically scan the organization's repositories for any repositories that contain a Jenkinsfile. If a Jenkinsfile is found, Jenkins will create a Multibranch Pipeline job for that repository.

Organization Folder Jobs can be used to:

- Automate the process of creating and managing Jenkins jobs for large organizations.
- Provide a centralized view of all the Jenkins jobs for an organization.
- Make it easier to manage and configure Jenkins jobs for an organization.

6) Create Jenkins pipeline for your repo and use script file (jenkinsfile) to write pipeline syntax, include parameter functions and env variable in it?



Dashboard > my-pipeline-project > Configuration

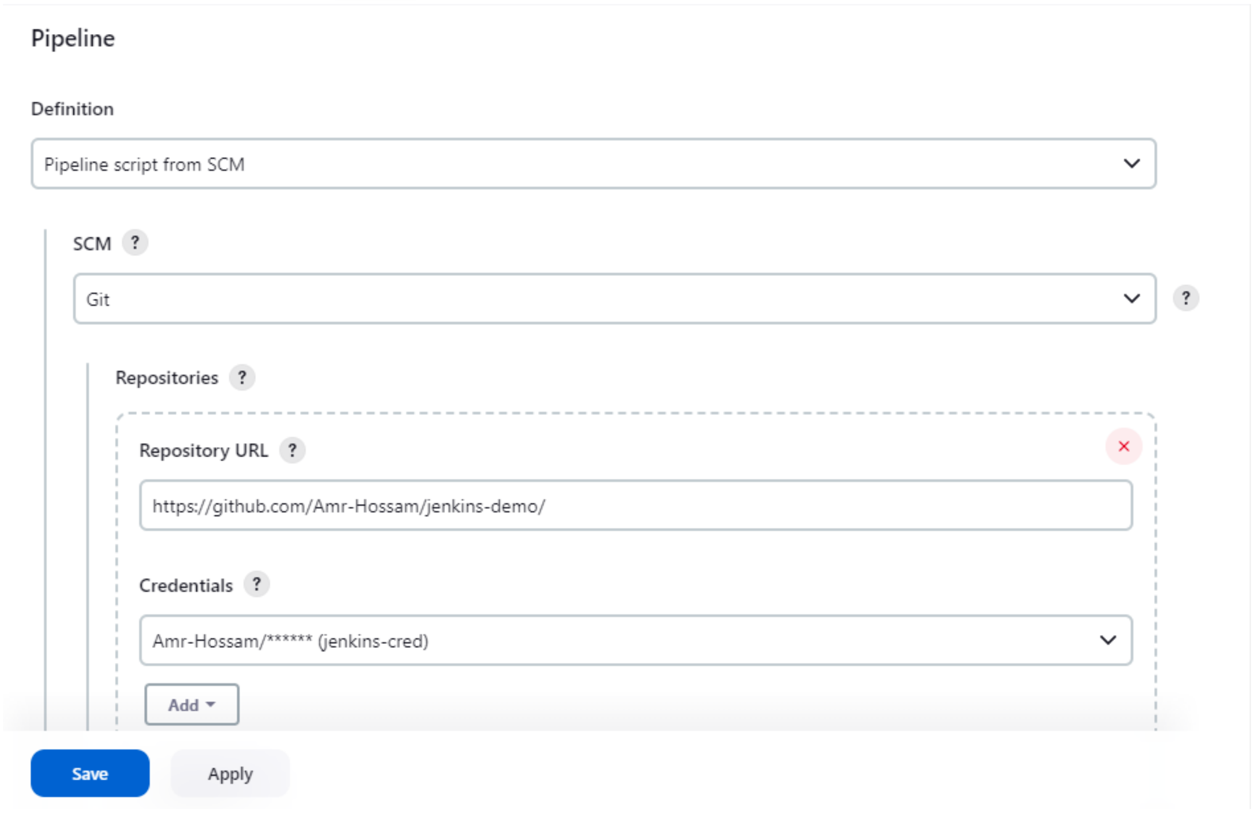
Configure

General
Advanced Project Options
Pipeline

✓ This project is parameterized

Boolean Parameter
Name: project
✓ Set by Default
Description: this paramater help you to know project name
[Plain text] Preview

Choice Parameter
Name: namespace
Choices:
dev
prod
stage



Pipeline

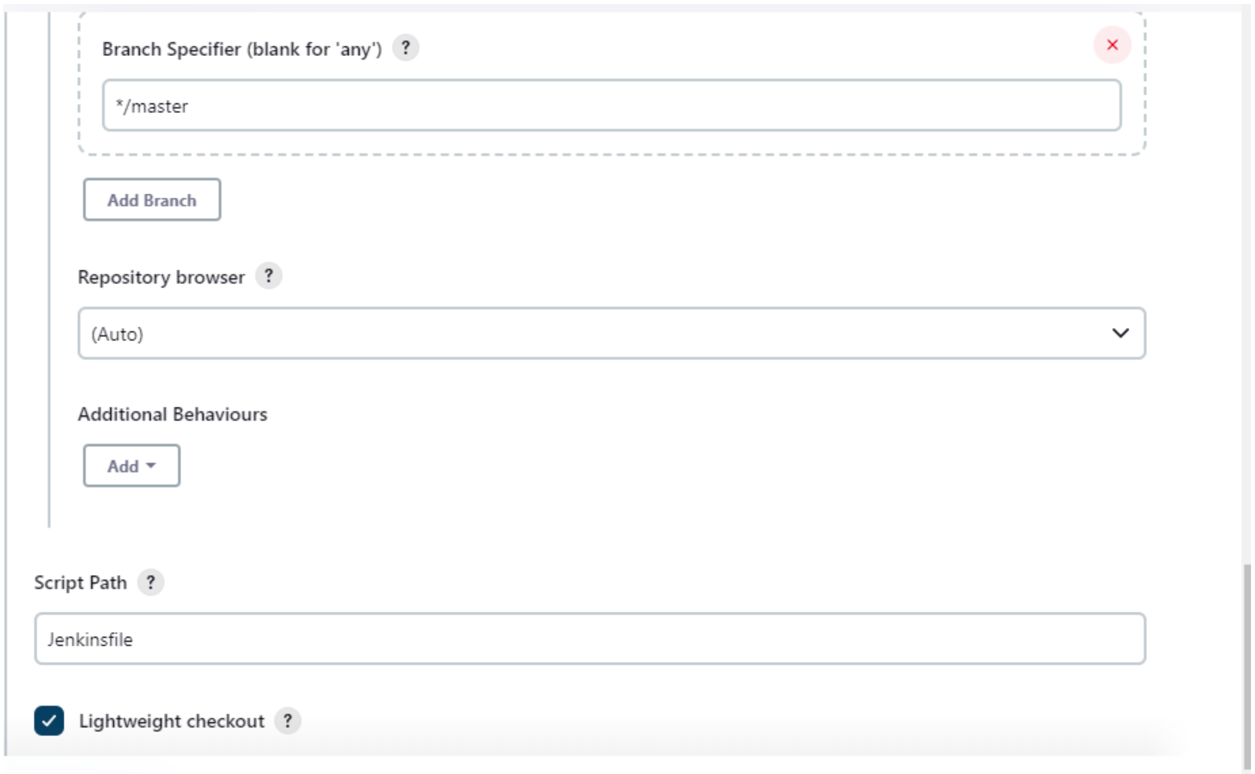
Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/Amr-Hossam/jenkins-demo/

Credentials: Amr-Hossam/****** (jenkins-cred)



Branch Specifier (blank for 'any')

*/master

Add Branch

Repository browser

(Auto)

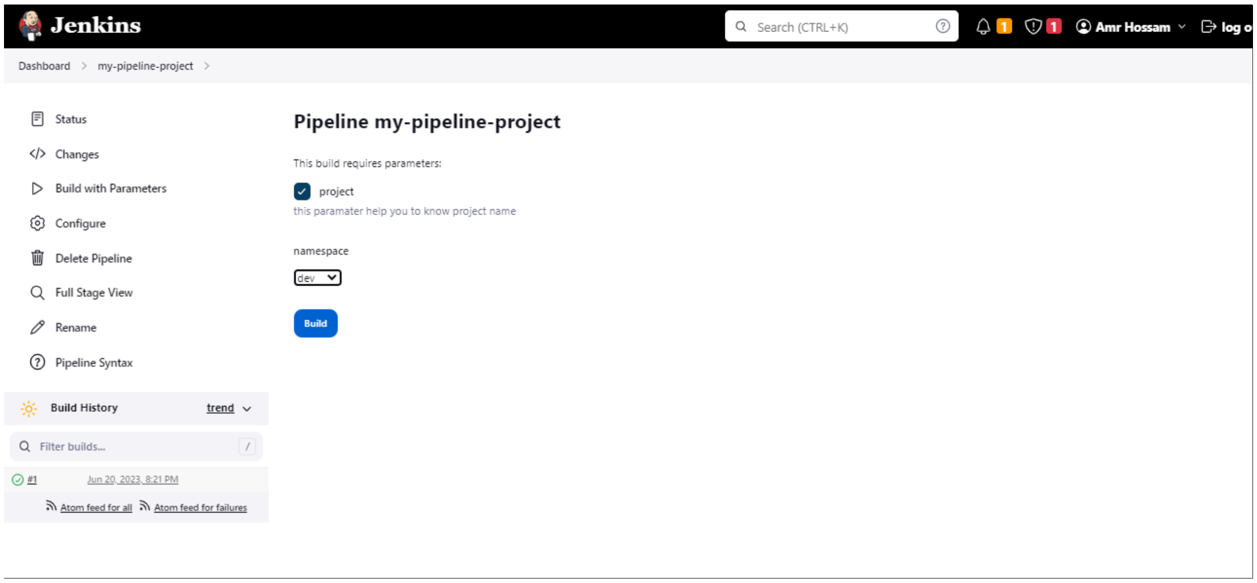
Additional Behaviours

Add

Script Path

Jenkinsfile

✓ Lightweight checkout



Jenkins

Dashboard > my-pipeline-project >

Pipeline my-pipeline-project

This build requires parameters:

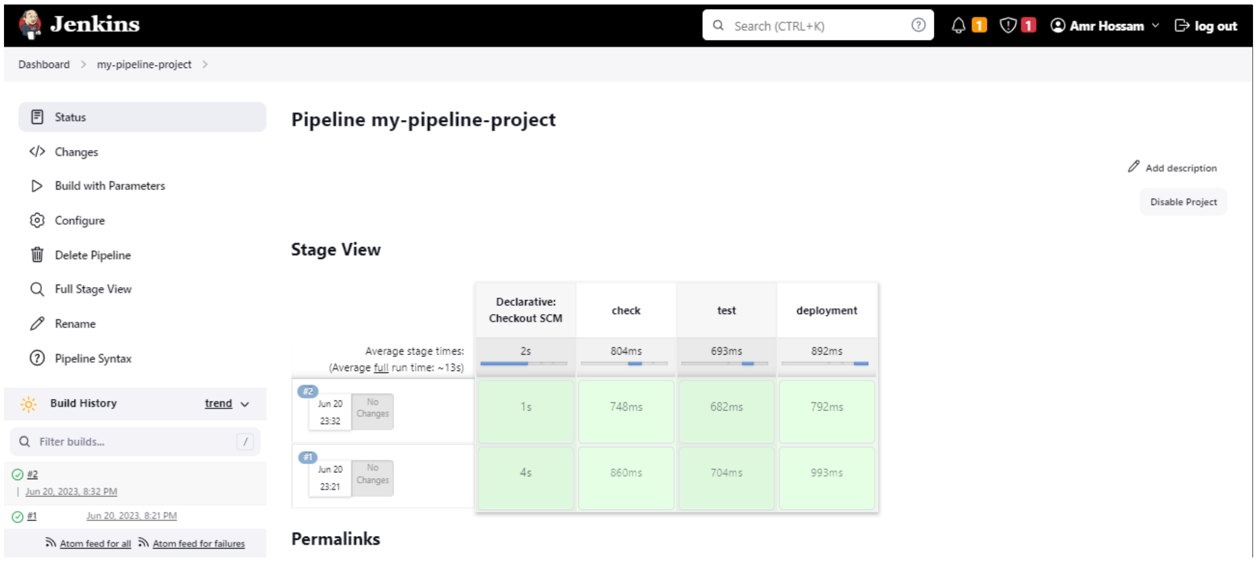
✓ project

this paramater help you to know project name

namespace

dev

Build

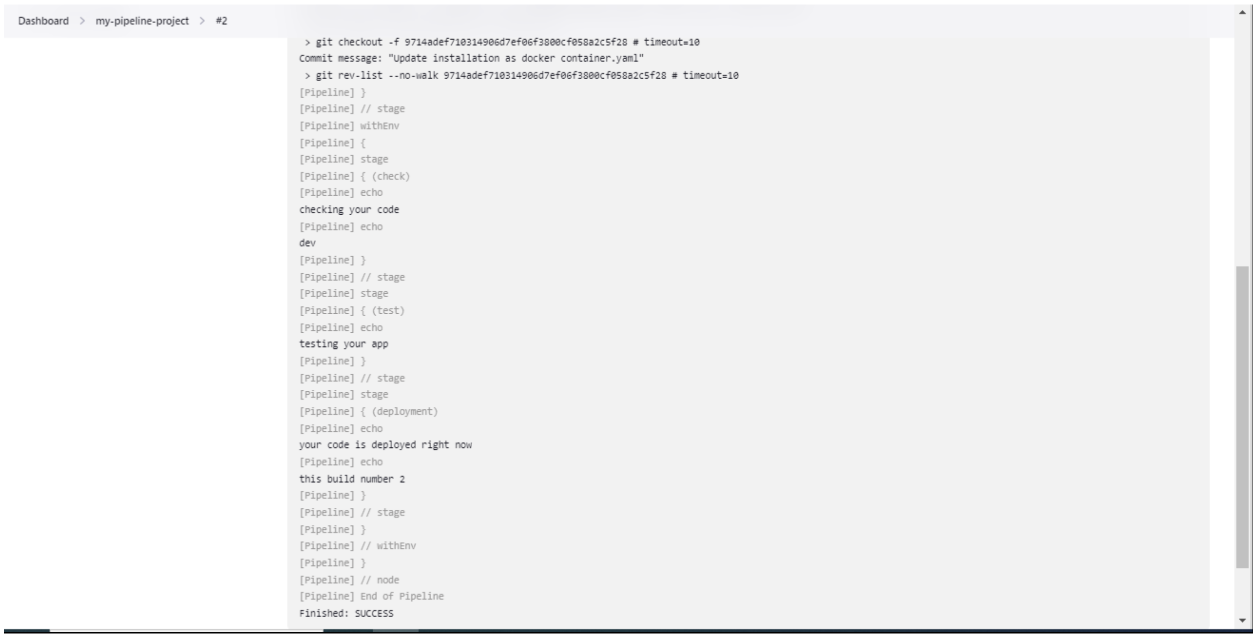


Pipeline my-pipeline-project

Stage View

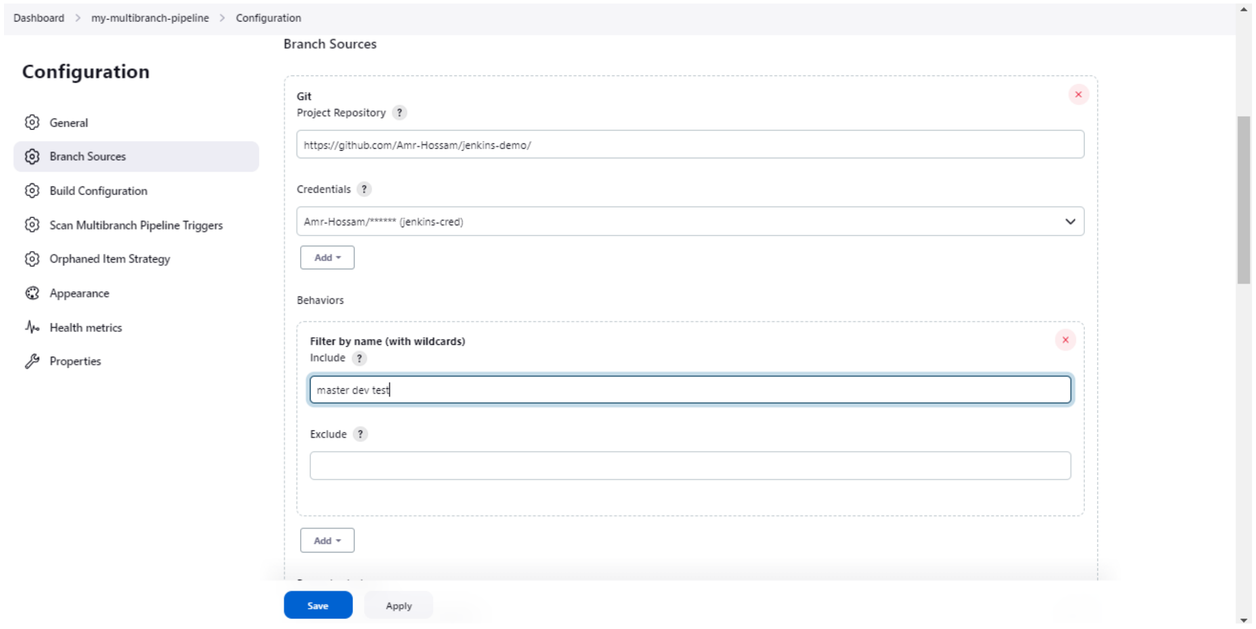
	Declarative: Checkout SCM	check	test	deployment
Average stage times: (Average full run time: ~13s)	2s	804ms	693ms	892ms
#2 Jun 20 23:32 No Changes	1s	748ms	682ms	792ms
#1 Jun 20 23:21 No Changes	4s	860ms	704ms	993ms

Permalinks



```
 > git checkout -f 9714adef710314906d7ef06f3800cf058a2c5f28 # timeout=10
Commit message: "Update installation as docker container.yaml"
 > git rev-list --no-walk 9714adef710314906d7ef06f3800cf058a2c5f28 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (check)
[Pipeline] echo
checking your code
[Pipeline] echo
dev
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (test)
[Pipeline] echo
testing your app
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (deployment)
[Pipeline] echo
your code is deployed right now
[Pipeline] echo
this build number 2
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

7) Create another multibranch pipeline and filter branches to contain only (master, dev, test)?



Branch Sources

Git

Project Repository: https://github.com/Amr-Hossam/jenkins-demo/

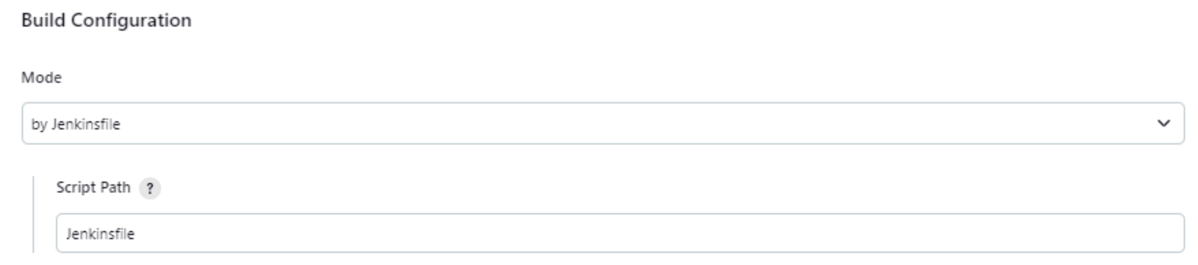
Credentials: Amr-Hossam/****** (jenkins-cred)

Behaviors

Filter by name (with wildcards)

Include: master dev test

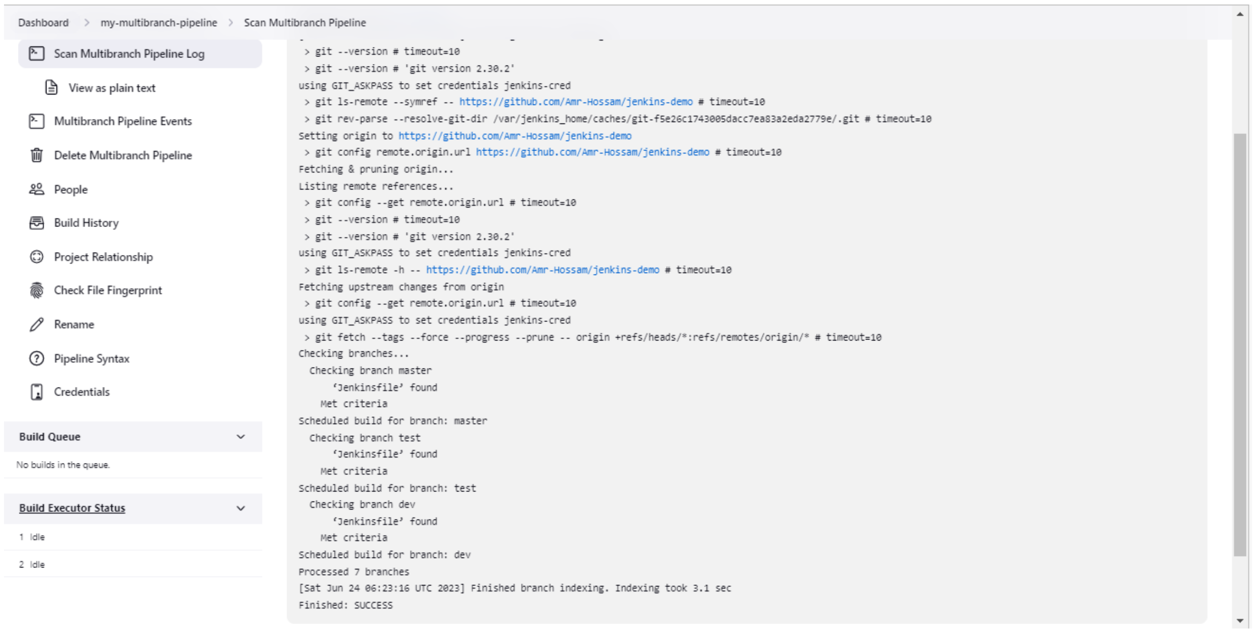
Exclude:



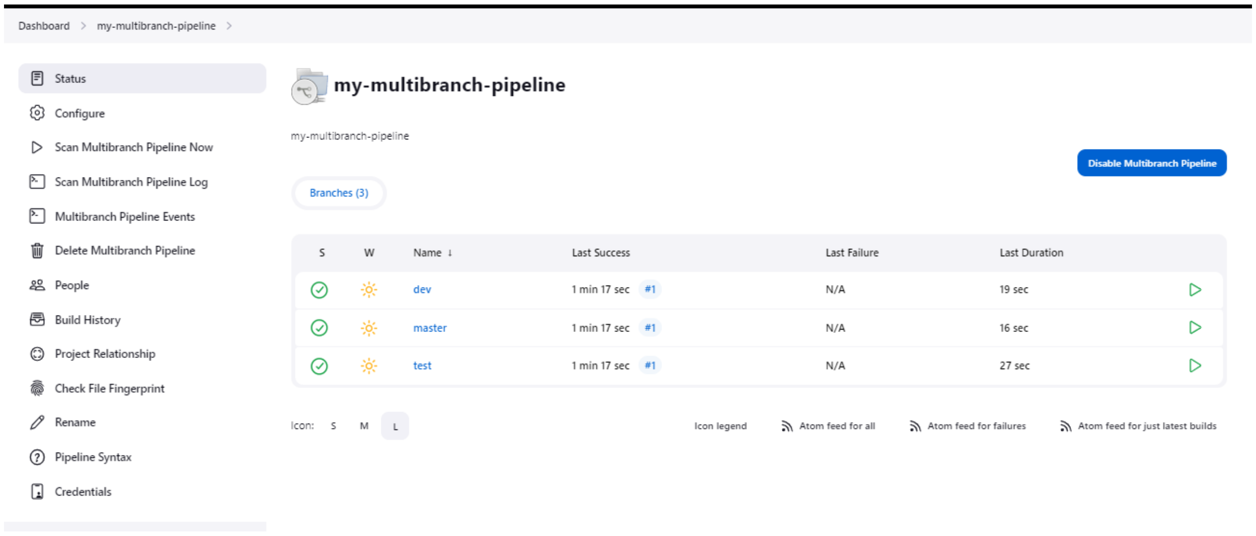
Build Configuration

Mode: by Jenkinsfile

Script Path: Jenkinsfile



```
 > git --version # timeout=10
 > git --version # 'git version 2.30.2'
using GIT_ASKPASS to set credentials jenkins-cred
 > git ls-remote --symref -- https://github.com/Amr-Hossam/jenkins-demo # timeout=10
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-f5e26c1743005dacc7ea83a2eda2779e/.git # timeout=10
Setting origin to https://github.com/Amr-Hossam/jenkins-demo
 > git config remote.origin.url https://github.com/Amr-Hossam/jenkins-demo # timeout=10
Fetching & pruning origin...
Listing remote references...
 > git config --get remote.origin.url # timeout=10
 > git --version # timeout=10
 > git --version # 'git version 2.30.2'
using GIT_ASKPASS to set credentials jenkins-cred
 > git ls-remote -h -- https://github.com/Amr-Hossam/jenkins-demo # timeout=10
Fetching upstream changes from origin
 > git config --get remote.origin.url # timeout=10
using GIT_ASKPASS to set credentials jenkins-cred
 > git fetch --tags --force --progress --prune -- origin +refs/heads/*:refs/remotes/origin/* # timeout=10
Checking branches...
  Checking branch master
      'Jenkinsfile' found
    Met criteria
Scheduled build for branch: master
  Checking branch test
      'Jenkinsfile' found
    Met criteria
Scheduled build for branch: test
  Checking branch dev
      'Jenkinsfile' found
    Met criteria
Scheduled build for branch: dev
Processed 7 branches
[Sat Jun 24 06:23:16 UTC 2023] Finished branch indexing. Indexing took 3.1 sec
Finished: SUCCESS
```



Dashboard > my-multibranch-pipeline >

- Status
- Configure
- Scan Multibranch Pipeline Now
- Scan Multibranch Pipeline Log
- Multibranch Pipeline Events
- Delete Multibranch Pipeline
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Rename
- Pipeline Syntax
- Credentials

my-multibranch-pipeline

my-multibranch-pipeline

Disable Multibranch Pipeline

Branches (3)

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	dev	1 min 17 sec #1	N/A	19 sec	▷
✓	☀	master	1 min 17 sec #1	N/A	16 sec	▷
✓	☀	test	1 min 17 sec #1	N/A	27 sec	▷

Icon: S M L

Icon legend · Atom feed for all · Atom feed for failures · Atom feed for just latest builds