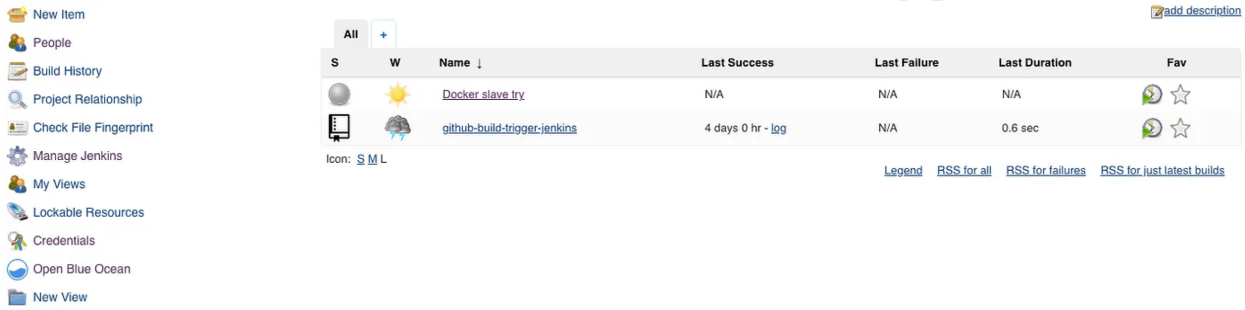
Integrate Jenkins with Github (Public repo)

One of the simple and interesting tools in DevOps is Jenkins. Another tool which is generally used is Github, which has one of the biggest open source community, is no more required to explain. There are a lot of interesting facts about Jenkins and one of them is that there are multiple ways you can install Jenkins but the best way is using the *command line installation*.

Prerequisite:

Jenkins should be installed and the initial Jenkins setup should be done, such that you can see the Jenkins dashboard. Your dashboard will look a bit different from mine as I have already created a Jenkins job and installed a few plugins which you can see on the left side of the image below.



- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Lockable Resources
- Credentials
- Open Blue Ocean
- New View

add description

All | +

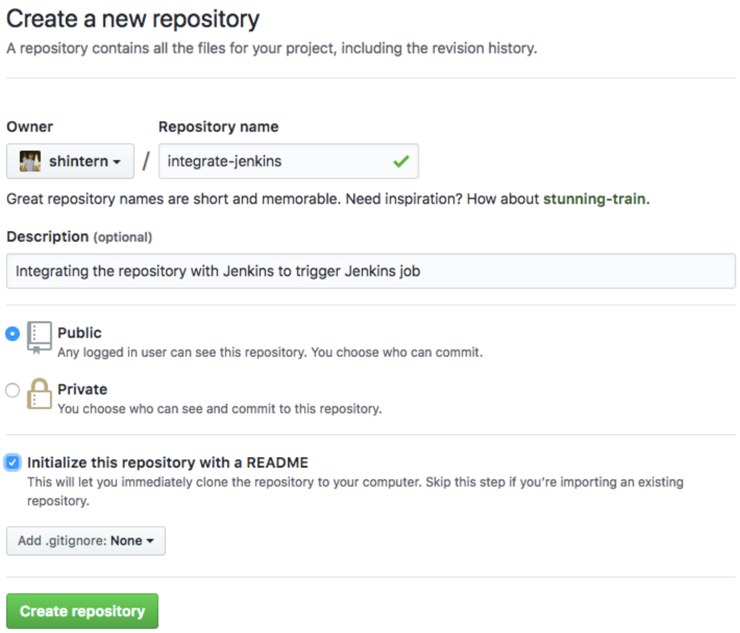
S	W	Name ↓	Last Success	Last Failure	Last Duration	Fav
		Docker slave try	N/A	N/A	N/A	☆
		github-build-trigger-jenkins	4 days 0 hr - log	N/A	0.6 sec	☆

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

Jenkins setup for Public Repository:

Open your Github profile and click on creating a new repository.



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: shintern / Repository name: integrate-jenkins ✓

Great repository names are short and memorable. Need inspiration? How about **stunning-train**.

Description (optional): Integrating the repository with Jenkins to trigger Jenkins job

● **Public**
Any logged in user can see this repository. You choose who can commit.

○ **Private**
You choose who can see and commit to this repository.

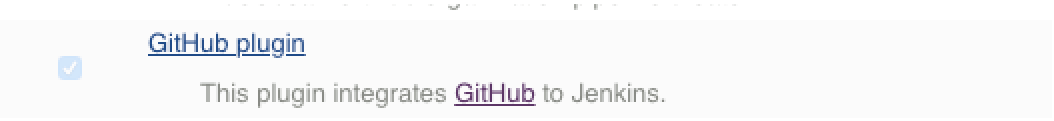
☑ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None

Creating a Github Repository

After you are done putting above things while creating the repository click on Create repository. A new repository will get created.

Install the Github plugin from the Manage plugins inside Jenkins. Go to Manage Jenkins -> Manage Plugins -> Available/ Installed. Check the Github Integration Plugin after clicking on Installed Plugins. If you are not able to find it check-in available maybe it is already installed due to the initial setup.

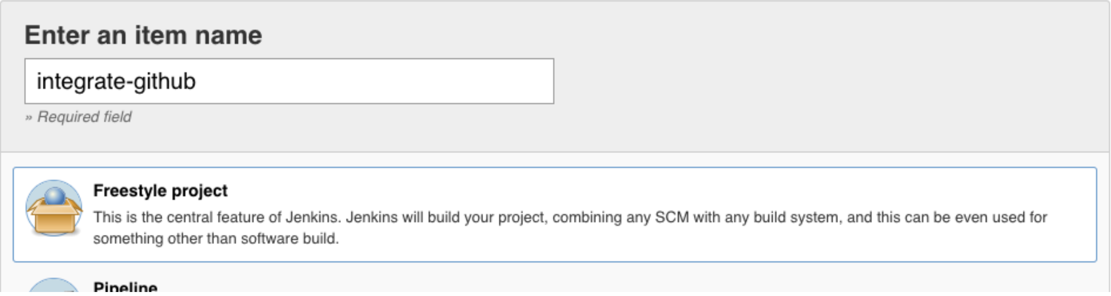


☑ GitHub plugin

This plugin integrates GitHub to Jenkins.

Github Integration Plugin

Click on new item -> Freestyle project -> ok.



Enter an item name

integrate-github

» Required field

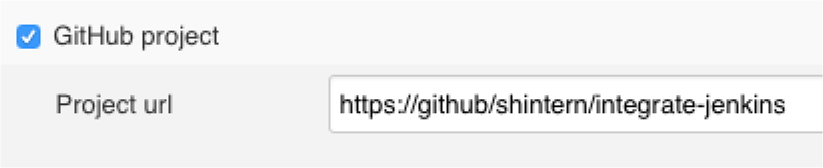
Freestyle project

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

Freestyle project

Go to General -> Github Project. Give your project URL

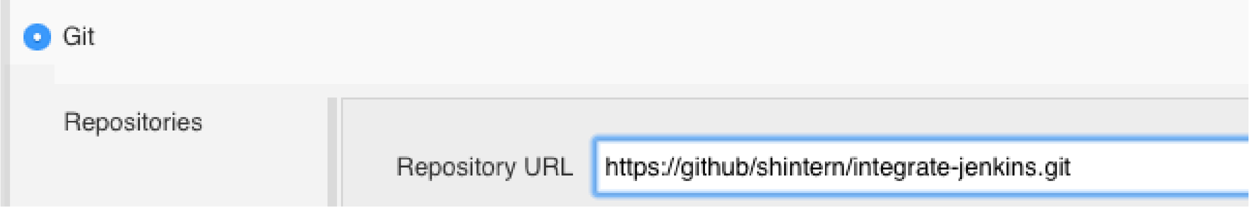


☑ GitHub project

Project url: https://github/shintern/integrate-jenkins

Github Project URL

Source Code Management -> Git. By default, it will take the master branch.

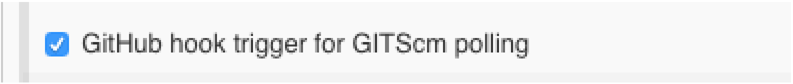


● Git

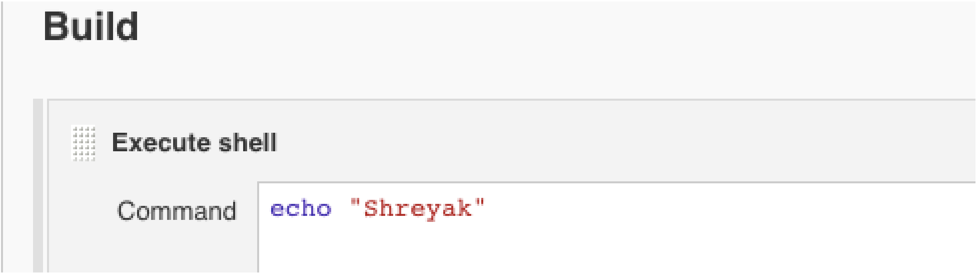
Repositories

Repository URL: https://github/shintern/integrate-jenkins.git

Source Code Management
Build Triggers -> GitHub hook trigger for GITScm polling


☑ GitHub hook trigger for GITScm polling

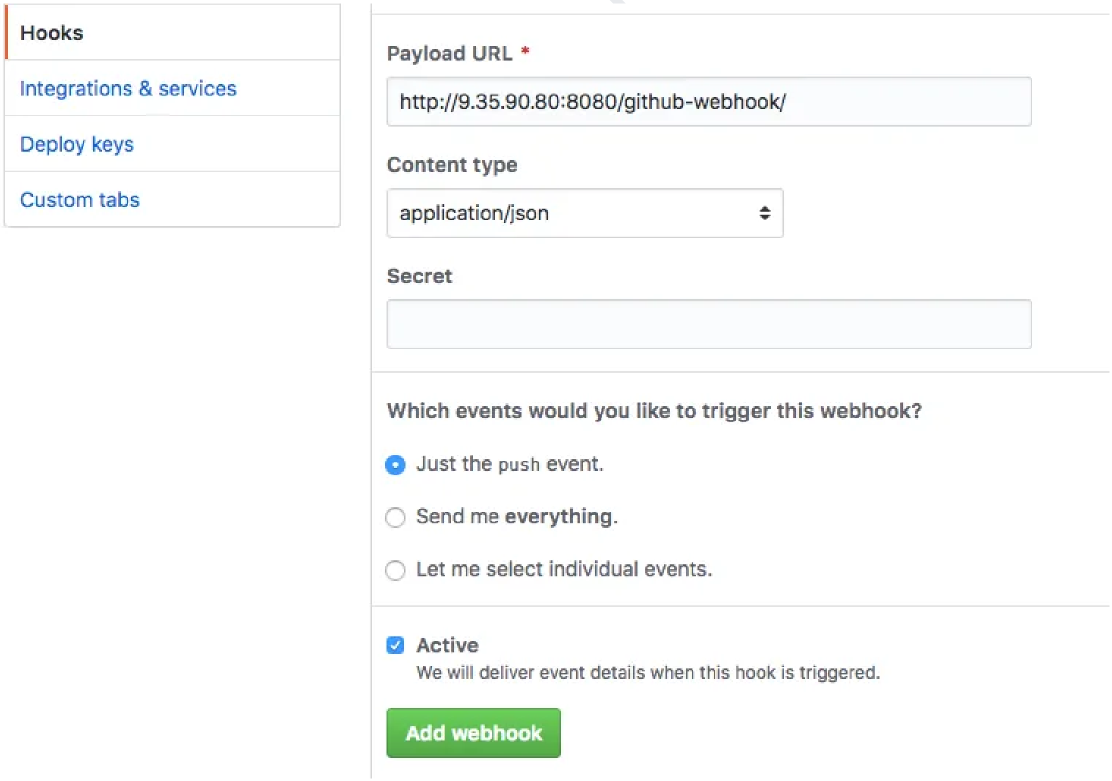
Build Triggers
Build -> Execute Shell. Put echo “Shreyak” inside the text input and press Save.


Build

Execute shell

Command
```
echo "Shreyak"
```

Github setup for Public Repository:
Open your Github repository which you created and go to settings. Click on Hooks and press Add Webhook. After you are done with this follow the below image configuration and press Add webhook (green button). Now the Github webhook is also set. This configuration works if your Jenkins has a public IP address. Otherwise, use [ngrok]() to create one for your localhost.



Payload URL *
http://9.35.90.80:8080/github-webhook/

Content type
application/json

Secret

Which events would you like to trigger this webhook?

● Just the push event.

○ Send me everything.

○ Let me select individual events.

☑ Active
We will deliver event details when this hook is triggered.

Yeah, that’s petty much it. Now make a commit to your Github Repository and you will be able to see the Jenkins job getting triggered. Try it out and understand everything by implementing all the steps at your pace.

Integrate Jenkins with Github (Private repo)

Connecting Jenkins to a private Github repository can be a pain if not done properly. Also, I would recommend you to check out how things work for a public Github repo as explained in the above section.

Prerequisite:

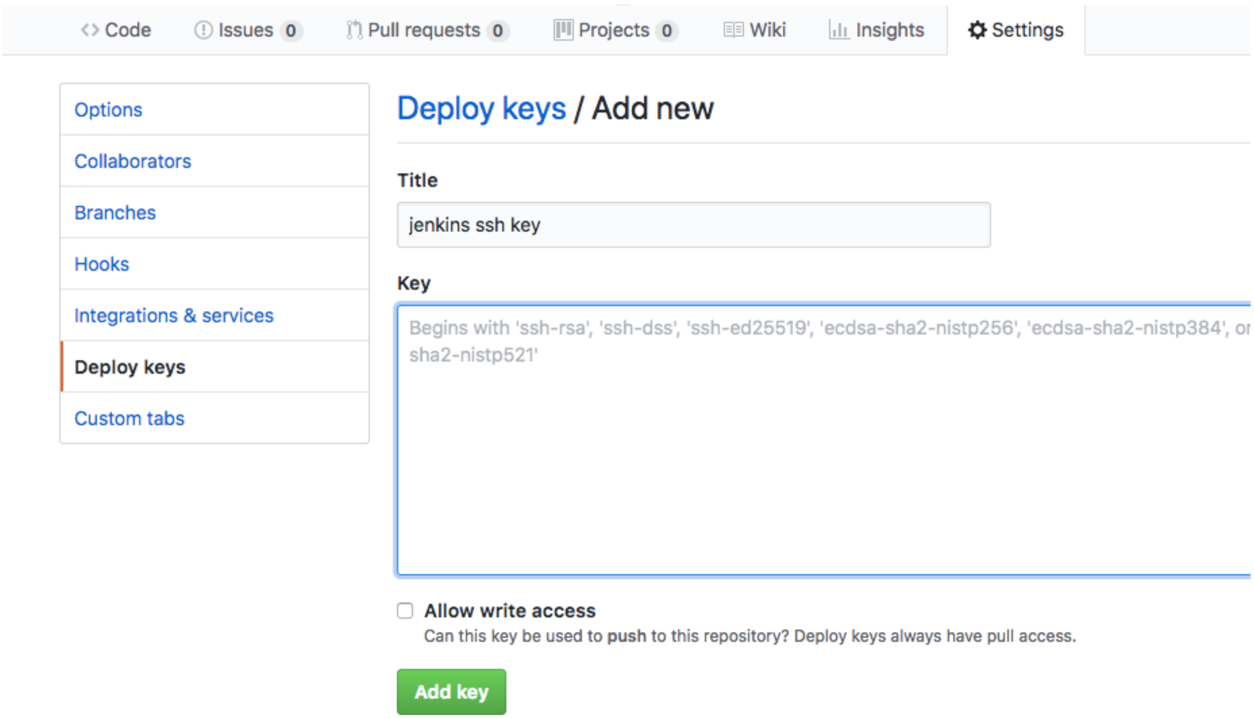
- Jenkins should be installed and the initial Jenkins setup needs to be done.
- A Private Github repository.

Generate ssh keys and add to Github:

For this task, we will be using ssh keys to connect to a private Github repository. You can also use username and password, later in this lab work, I will explain where you need to change instead of ssh keys for login credentials.

- Generate your ssh keys for Github.
- Adding generated keys to the private Github repository created before.

Go to your private Github repo -> Settings -> Deploy keys.



Deploy keys / Add new

Title

jenkins ssh key

Key

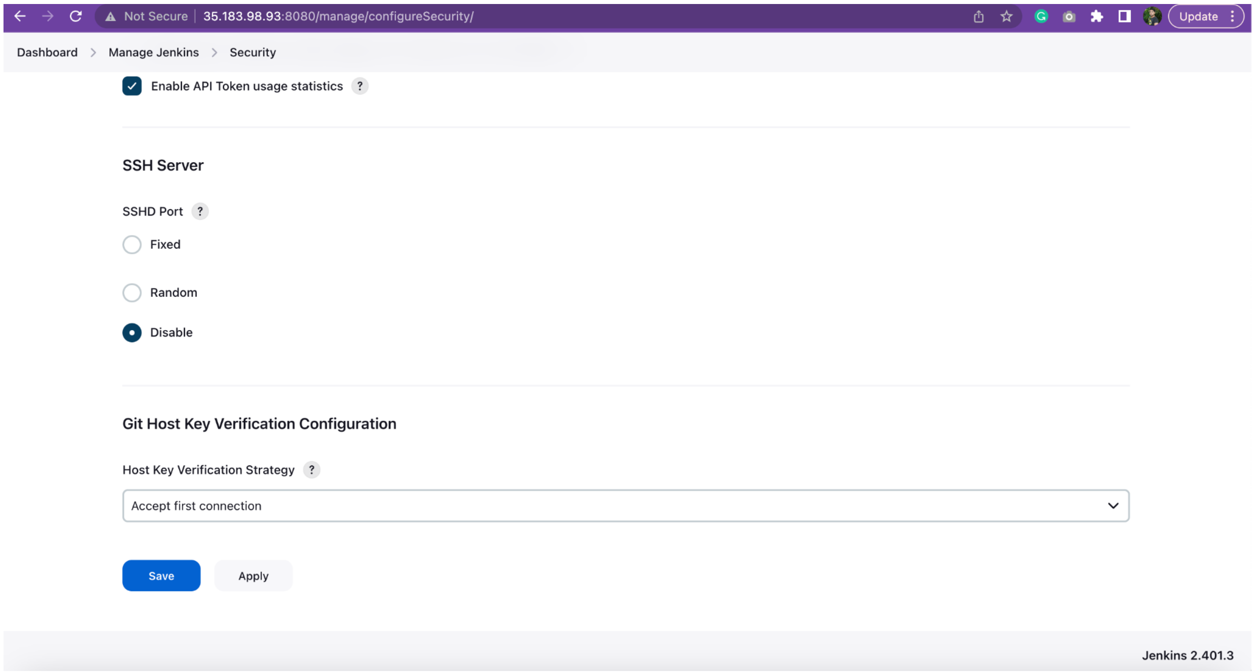
Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or sha2-nistp521'

□ Allow write access

Can this key be used to **push** to this repository? Deploy keys always have pull access.

Add ssh key to repo

First make sure the **Git Host Key Verification Configuration** is set to → **Accept first connection** by navigating to **Dashboard** → **Manage Jenkins** → **Security** menu. Refer the screenshot as added below:



Not Secure | 35.183.98.93:8080/manage/configureSecurity/

Dashboard > Manage Jenkins > Security

☑ Enable API Token usage statistics ?

SSH Server

SSHD Port ?

○ Fixed

○ Random

● Disable

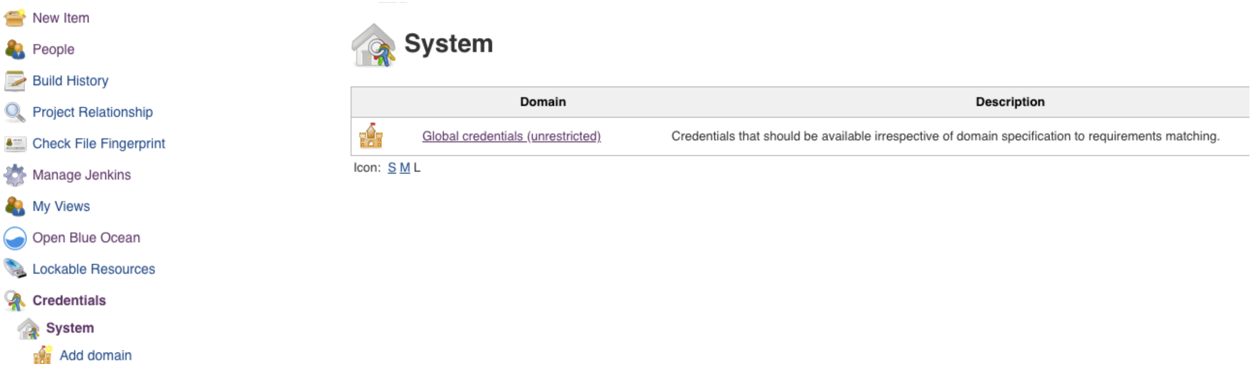
Git Host Key Verification Configuration

Host Key Verification Strategy ?

Accept first connection

Add your ssh key inside the key text input folder, if you want write access to the repo click on the Allow write access.

Jenkins configuration to access private repo:
Go to Jenkins dashboard -> Credentials -> System -> Global credentials -> Add credentials.

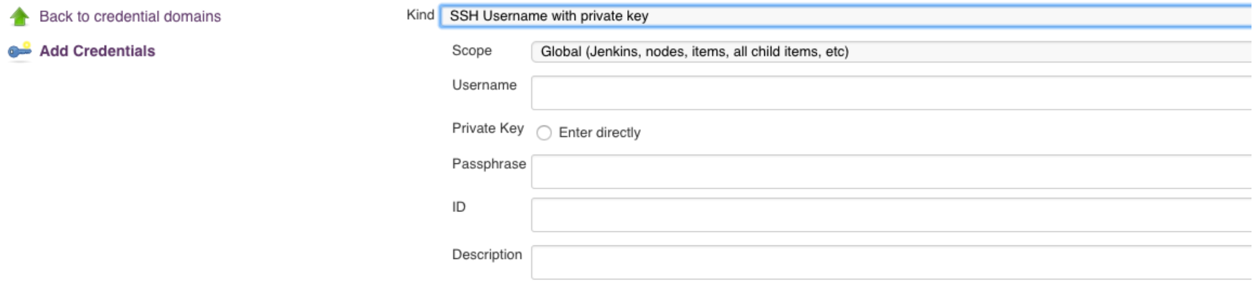


System

Domain	Description
Global credentials (unrestricted)	Credentials that should be available irrespective of domain specification to requirements matching.

Icon: S M L

Add new credential to Jenkins



Kind: SSH Username with private key

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

Username:

Private Key: ○ Enter directly

Passphrase:

ID:

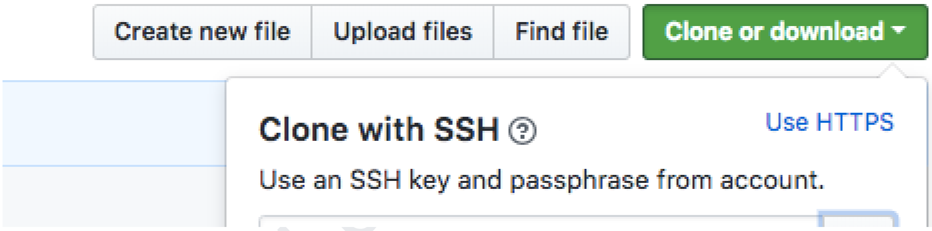
Description:

Add ssh username with a private key

Give username as Jenkins or whatever you can remember. Add the Private key -> Enter directly and copy paste the same ssh keys here, click on okay. Here you can select Username with password option if you want to use a password to access the private repo from Jenkins. As I mentioned at the beginning of tutorial that which step needs to be changed if you want to enable login with a password.

Till here we are done with Jenkins and Github configuration setup. Now set up the Jenkins job to take this private repo and trigger job on a new commit to the repository.
Jenkins job setup:

Create a new Jenkins job from the dashboard. New item -> Freestyle Project. After this go to source code management -> git. Copy the ssh URL of your Github repo. As you have used ssh key you need to use ssh URL instead of https URL.

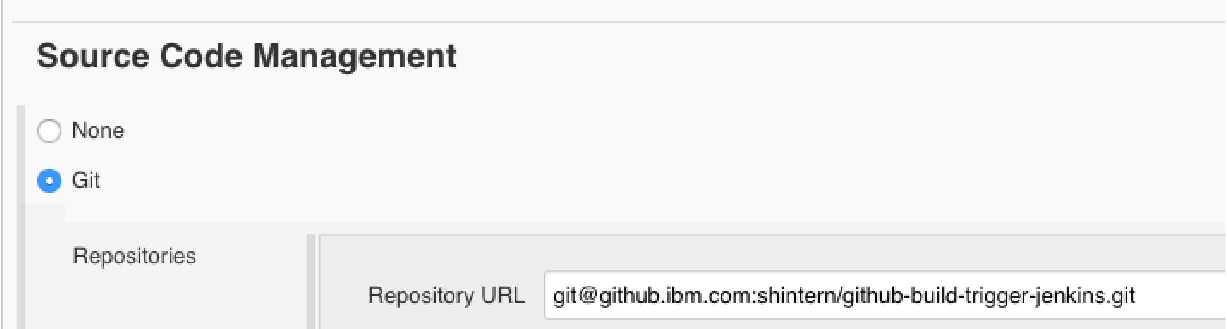


Clone with SSH

Use an SSH key and passphrase from account.

SSH URL for the repo.

Go to your newly created job in Jenkins and select Git inside Source Code Mangement


Source Code Management

○ None

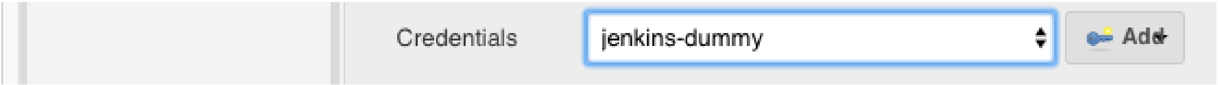
● Git

Repositories

Repository URL: git@github.ibm.com:shintern/github-build-trigger-jenkins.git

Paste ssh URL to source code management

Add the credentials you created before to the credentials section in source code management under the above Repository URL. I have used the name for those credentials as jenkins-dummy.



Credentials jenkins-dummy

That’s it. Save and try running your job.