Jenkins 101

Cheryl Fong

View on Google slides

Objective

Topics covered:

Installation Prerequisites

Minimum hardware requirements:

- 256 MB of RAM
- 1 GB of drive space (10 GB is a recommended minimum if running Jenkins as a Docker container)

Recommended hardware configuration for a *small team*:

- 1 GB+ of RAM
- 50 GB+ of drive space

Software requirements:

- Java: see the Java Requirements page
- Web browser: see the Web Browser Compatibility page

Jenkins Official Prerequisites Guide

Installation

Installation and setup on your machine:

- See the official Jenkins guides for:
   i. MacOS
   ii. Linux distros - Fedora and Debian/Ubuntu
   iii. Windows

Note: It is recommended to install Jenkins on a provisioned server, not on a personal computer. So that you could run Jenkins 24/7 and would be able to access the Jenkins dashboard via a public IP address.

Installation and Setup via Docker are covered in the following slides.

Installation via Docker

⚠️

This section assumes you have Docker up and running.

Note: your system must meet the "Minimum hardware requirements" but you may ignore the "Software requirements" on slide 3.

Otherwise Install Docker CE Edition on MacOS, Linux or Windows before proceeding

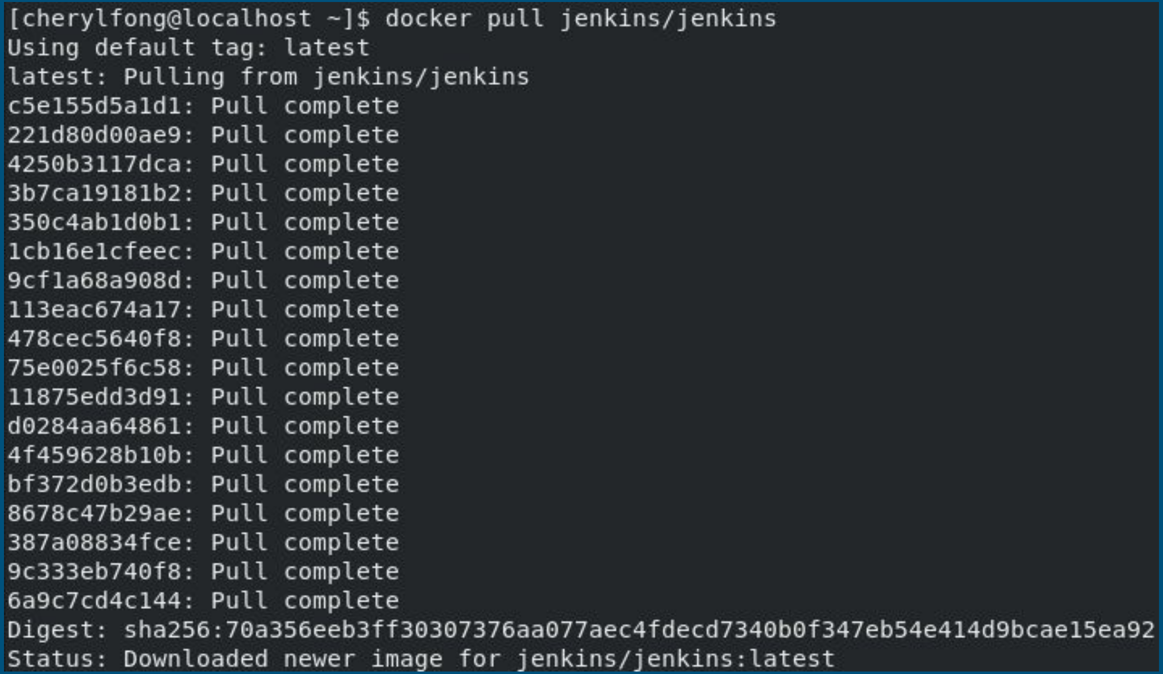
If you've installed Jenkins natively (not through Docker), skip to slide: 15

Installation via Docker

Run the following command on your terminal to pull the latest official jenkins container image:

```
docker pull jenkins/jenkins
```

Installation via Docker



```
[cherylfong@localhost ~]$ docker pull jenkins/jenkins
Using default tag: latest
latest: Pulling from jenkins/jenkins
c5e155d5a1d1: Pull complete
221d80d00ae9: Pull complete
4250b3117dca: Pull complete
3b7ca19181b2: Pull complete
350c4ab1d0b1: Pull complete
1cb16e1cfeec: Pull complete
9cf1a68a908d: Pull complete
113eac674a17: Pull complete
478cec5640f8: Pull complete
75e0025f6c58: Pull complete
11875edd3d91: Pull complete
d0284aa64861: Pull complete
4f459628b10b: Pull complete
bf372d0b3edb: Pull complete
8678c47b29ae: Pull complete
387a08834fce: Pull complete
9c333eb740f8: Pull complete
6a9c7cd4c144: Pull complete
Digest: sha256:70a356eeb3ff30307376aa077aec4fdecd7340b0f347eb54e414d9bcae15ea92
Status: Downloaded newer image for jenkins/jenkins:latest
```

Installation via Docker

This is the command required to get Jenkins running, see the next slide before running this command:

```
docker run -d -v jenkins_home:/var/jenkins_home -p 8080:8080 -p 50000:50000 jenkins/jenkins
```

Setup via Docker

```
docker run -d -v jenkins_home:/var/jenkins_home -p 8080:8080 -p 50000:50000 jenkins/jenkins
```

The command explained:

1. Runs the Jenkins image container in detached mode *-d*
2. Connects the container to a volume named `jenkins_home` and attach it to the container's `/var/jenkins_home directory.` (You can find the *jenkins_home* volume by running `Docker volume ls`)
3. *-p* specifies connecting the host 8080 port to the container's 8080 port
4. Read the Build Executors section on the usage of `-p 50000:50000`. It is likely that you'll not need this. Hence the following is sufficient:

```
docker run -d -v jenkins_home:/var/jenkins_home -p 8080:8080 jenkins/jenkins
```

Official Jenkins Docker Container Documentation

Setup via Docker (Advanced)

Use case:

You have a website running on some IP address on port 8080 e.g. `13.52.51.45:8080` but you want to bind Jenkins to the same IP address.

1. Find an unoccupied port on the host/server e.g. 8082 (or 80 if nothing is bounded to it).
2. Bind the Jenkins container to the host 8082 port to its 8080 port, with the following command:

```
docker run -d -v jenkins_home:/var/jenkins_home -p 8082:8080 jenkins/jenkins
```

This allows you to visit the Jenkins Dashboard at *`http://13.52.51.45:8082`*

Installation via Docker (Advanced++)

Use case:

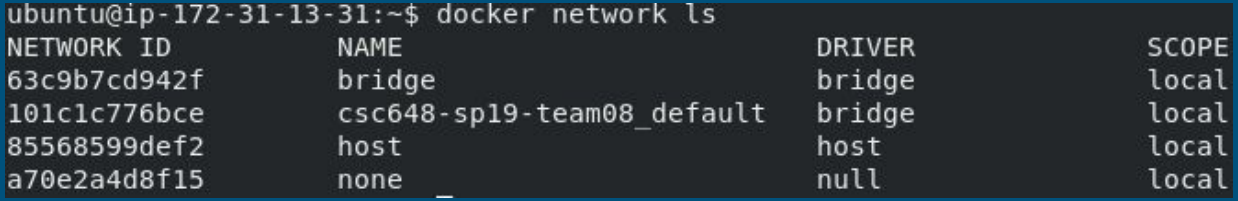
You have a website running on some IP address on port 8080 e.g. `13.52.51.45:8080` that you've launched via Docker Compose but you want to bind Jenkins to the same IP address.

1. Find an unoccupied port on the host/server e.g. 8082 or (80 if nothing is bounded to it).
2. Find the network namespace created by Docker Compose.
   a. Run: `docker network ls`
3. Bind the Jenkins container to the host 8082 port to its 8080 port, with the command on the next slide.

Setup via Docker (Advanced++)

Use case:

You have a website running on some IP address on port 8080 e.g. `13.52.51.45:8080` that you've launched via Docker Compose but you want to bind Jenkins to the same IP address.



```
ubuntu@ip-172-31-13-31:~$ docker network ls
NETWORK ID          NAME                          DRIVER              SCOPE
63c9b7cd942f        bridge                        bridge              local
101c1c776bce        csc648-sp19-team08_default    bridge              local
85568599def2        host                          host                local
a70e2a4d8f15        none                          null                local
```

```
docker run -d -v jenkins_home:/var/jenkins_home --net csc648-sp19-team08_default -p 8082:8080
jenkins/jenkins
```

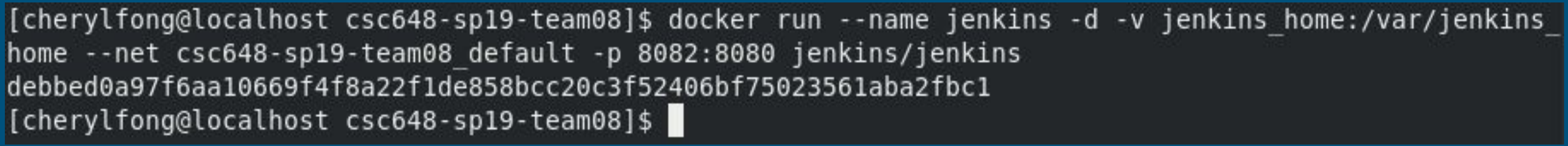
This allows you to visit the Jenkins Dashboard at *`http://13.52.51.45:8082`*

Setup via Docker (Advanced++)

Use case:

You have a website running on some IP address on port 8080 e.g. `13.52.51.45:8080` that you've launched via Docker Compose but you want to bind Jenkins to the same IP address.

```
docker run -d -v jenkins_home:/var/jenkins_home --net csc648-sp19-team08_default -p 8082:8080
jenkins/jenkins
```



```
[cherylfong@localhost csc648-sp19-team08]$ docker run --name jenkins -d -v jenkins_home:/var/jenkins_
home --net csc648-sp19-team08_default -p 8082:8080 jenkins/jenkins
debbed0a97f6aa10669f4f8a22f1de858bcc20c3f52406bf75023561aba2fbc1
[cherylfong@localhost csc648-sp19-team08]$
```

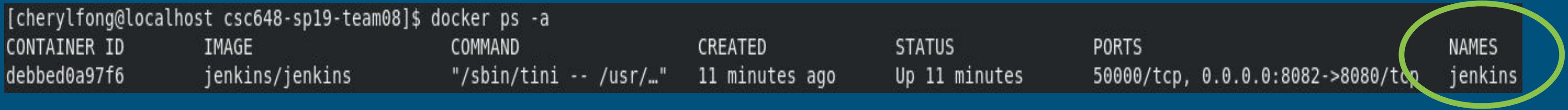
The expected output above shows a hash value, i.e. hash 'debb . . . fbc1'. It varies from execution of the run command in detached mode. Your hash value will not be the same as the screenshot.

Setup via Docker (Advanced++)

Use case:

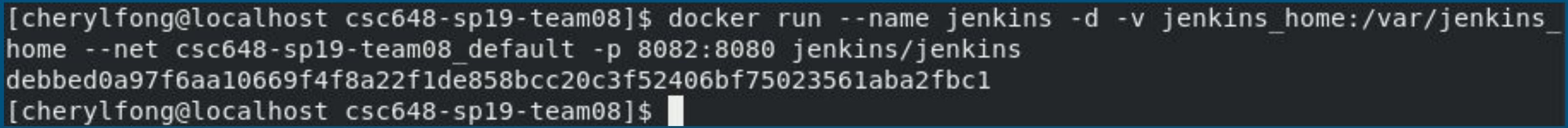
You have a website running on some IP address on port 8080 e.g. `13.52.51.45:8080` that you've launched via Docker Compose but you want to bind Jenkins to the same IP address.

You can verify that your jenkins container is running by executing, *`docker ps -a`* :



```
[cherylfong@localhost csc648-sp19-team08]$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                               NAMES
debbed0a97f6        jenkins/jenkins     "/sbin/tini -- /usr/…"   11 minutes ago      Up 11 minutes       50000/tcp, 0.0.0.0:8082->8080/tcp   jenkins
```

The *`--name`* option lets you provide a custom container name for the running Jenkins image container.



```
[cherylfong@localhost csc648-sp19-team08]$ docker run --name jenkins -d -v jenkins_home:/var/jenkins_
home --net csc648-sp19-team08_default -p 8082:8080 jenkins/jenkins
debbed0a97f6aa10669f4f8a22f1de858bcc20c3f52406bf75023561aba2fbc1
[cherylfong@localhost csc648-sp19-team08]$ 
```

In this case, I've named the Jenkins image container as 'jenkins'.

Setup

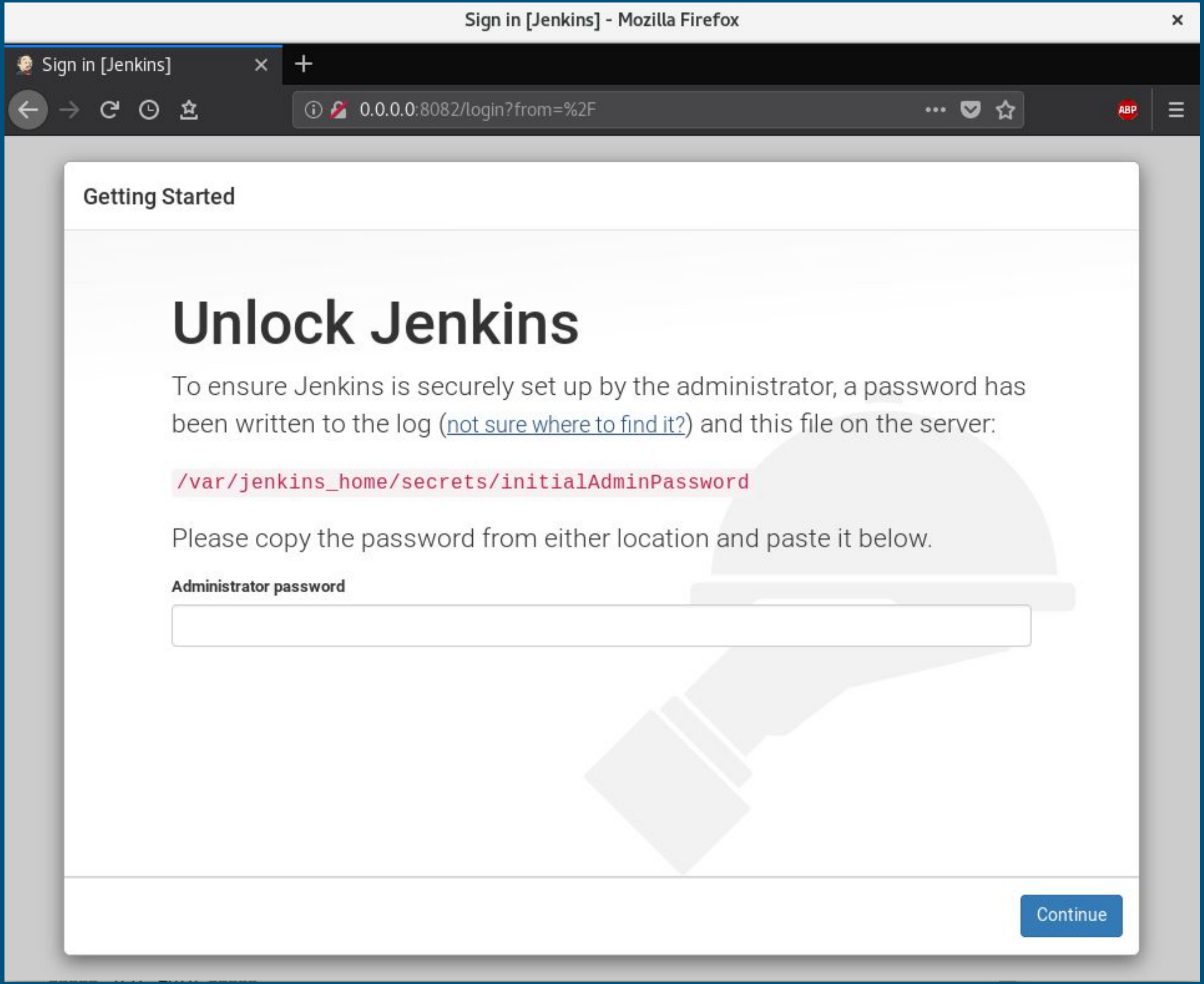
This "Setup" section will guide you through setting up Jenkins via the recommended/default configuration.

⚠️ At this point you should have Jenkins installed on your machine natively or via Docker.

Access the Jenkins Dashboard by the IP address of the machine you've installed it on:

IP Examples	Docker or Natively Installed	Advanced Installation via Docker (on slide 10)
On a server with an exposed public accessible IP	http://13.52.51.45:80 or http://13.52.51.45	http://13.52.51.45:8082
On your personal computer (not recommended see slide 4)	http://0.0.0.0:80 or http://0.0.0.0 http://127.0.0.1:80 or http://127.0.0.1	http://0.0.0.0:8082/ or http://127.0.0.1:8082/

Setup


Sign in [Jenkins] - Mozilla Firefox

Sign in [Jenkins]

0.0.0.0:8082/login?from=%2F

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Setup

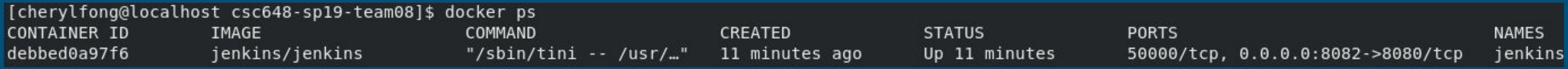
As instructed by Jenkins, copy and paste the initial administrator password into the field.

To obtain the password, SSH into your server and locate the `/var/jenkins_home/secrets/initialAdminPassword` file.

See the slide 18 to obtain the password from the Jenkins Image container.

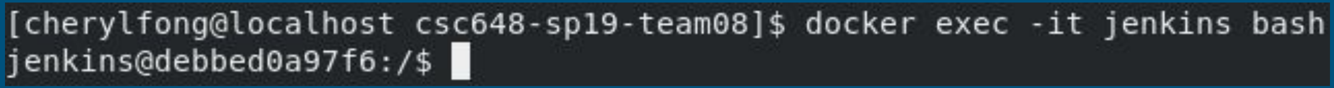
Setup

My Jenkins Image container that is running is named ‘jenkins’.



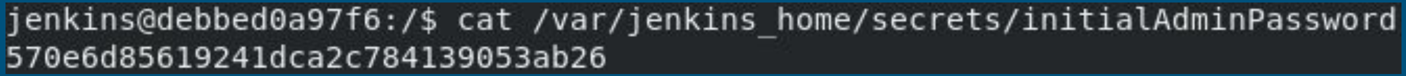
```
[cherylfong@localhost csc648-sp19-team08]$ docker ps
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS              PORTS                                NAMES
debbed0a97f6        jenkins/jenkins          "/sbin/tini -- /usr/…"   11 minutes ago      Up 11 minutes       50000/tcp, 0.0.0.0:8082->8080/tcp    jenkins
```

The following command will execute an interactive bash terminal into my ‘jenkins’ container:



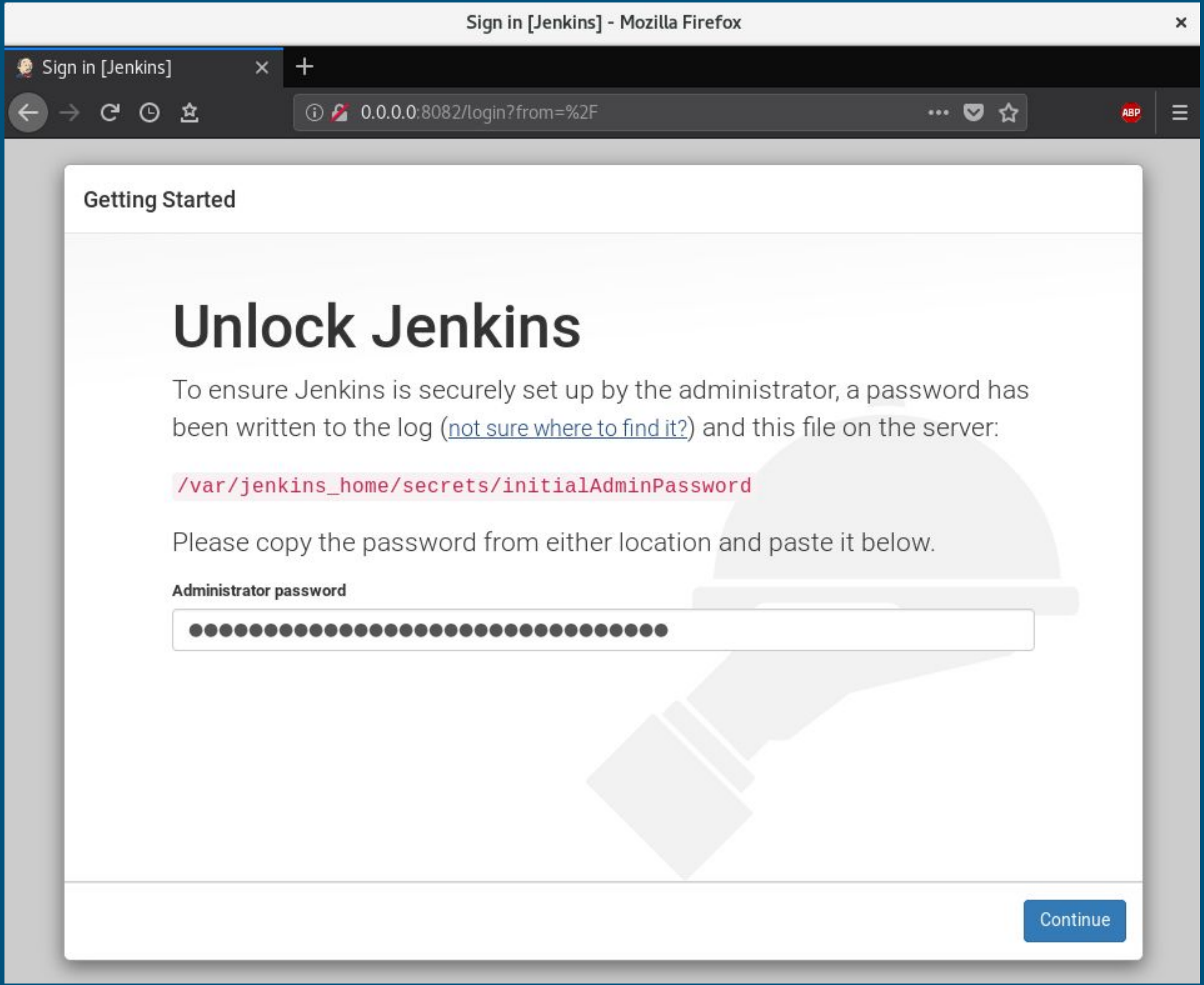
```
[cherylfong@localhost csc648-sp19-team08]$ docker exec -it jenkins bash
jenkins@debbed0a97f6:/$ █
```

Copy and paste the output from the command into the password field on the browser:



```
jenkins@debbed0a97f6:/$ cat /var/jenkins_home/secrets/initialAdminPassword
570e6d85619241dca2c784139053ab26
```

Setup


Sign in [Jenkins] - Mozilla Firefox

Sign in [Jenkins]

0.0.0.0:8082/login?from=%2F

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/jenkins_home/secrets/initialAdminPassword`

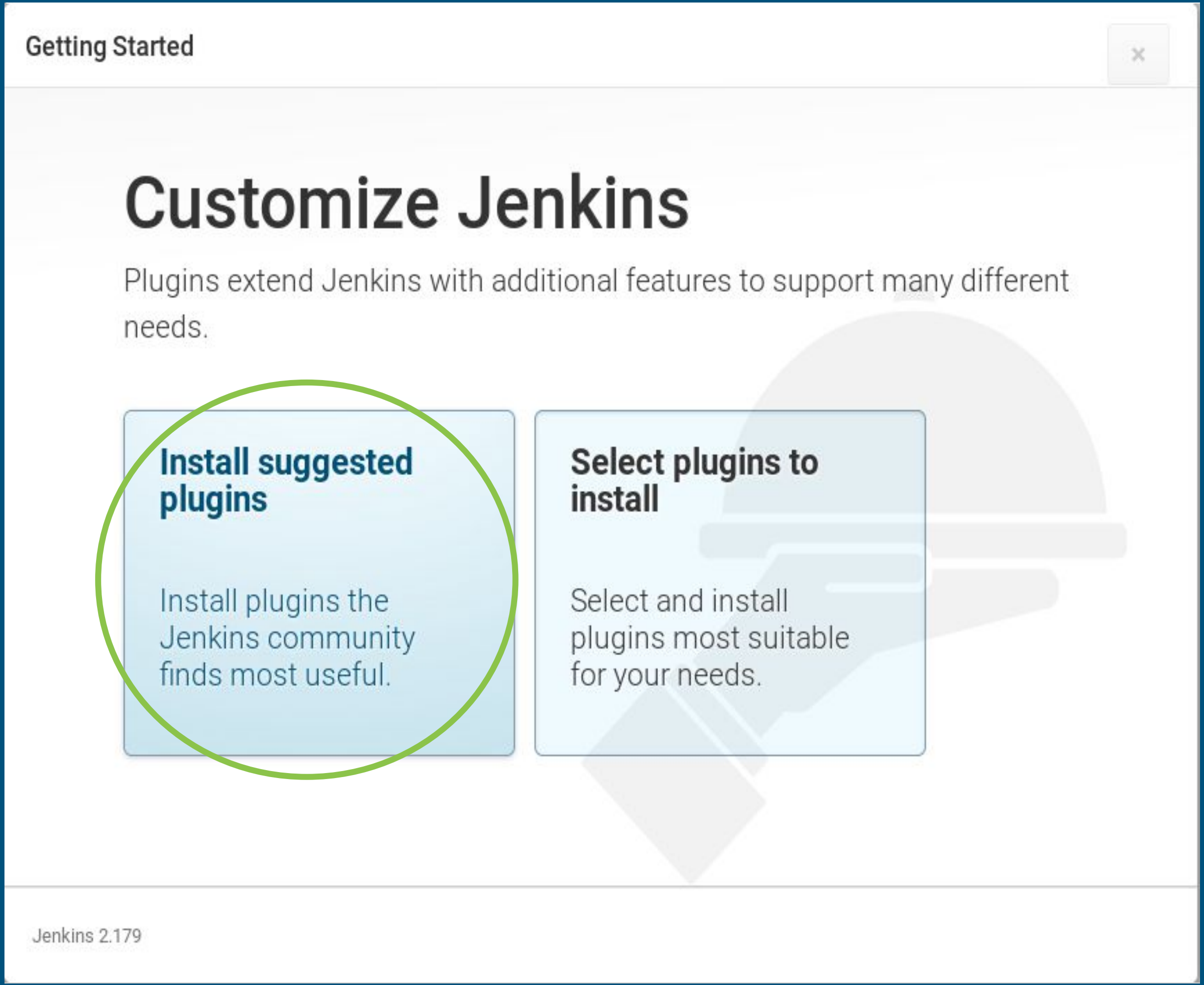
Please copy the password from either location and paste it below.

Administrator password

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

Continue

Setup



Getting Started

×

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

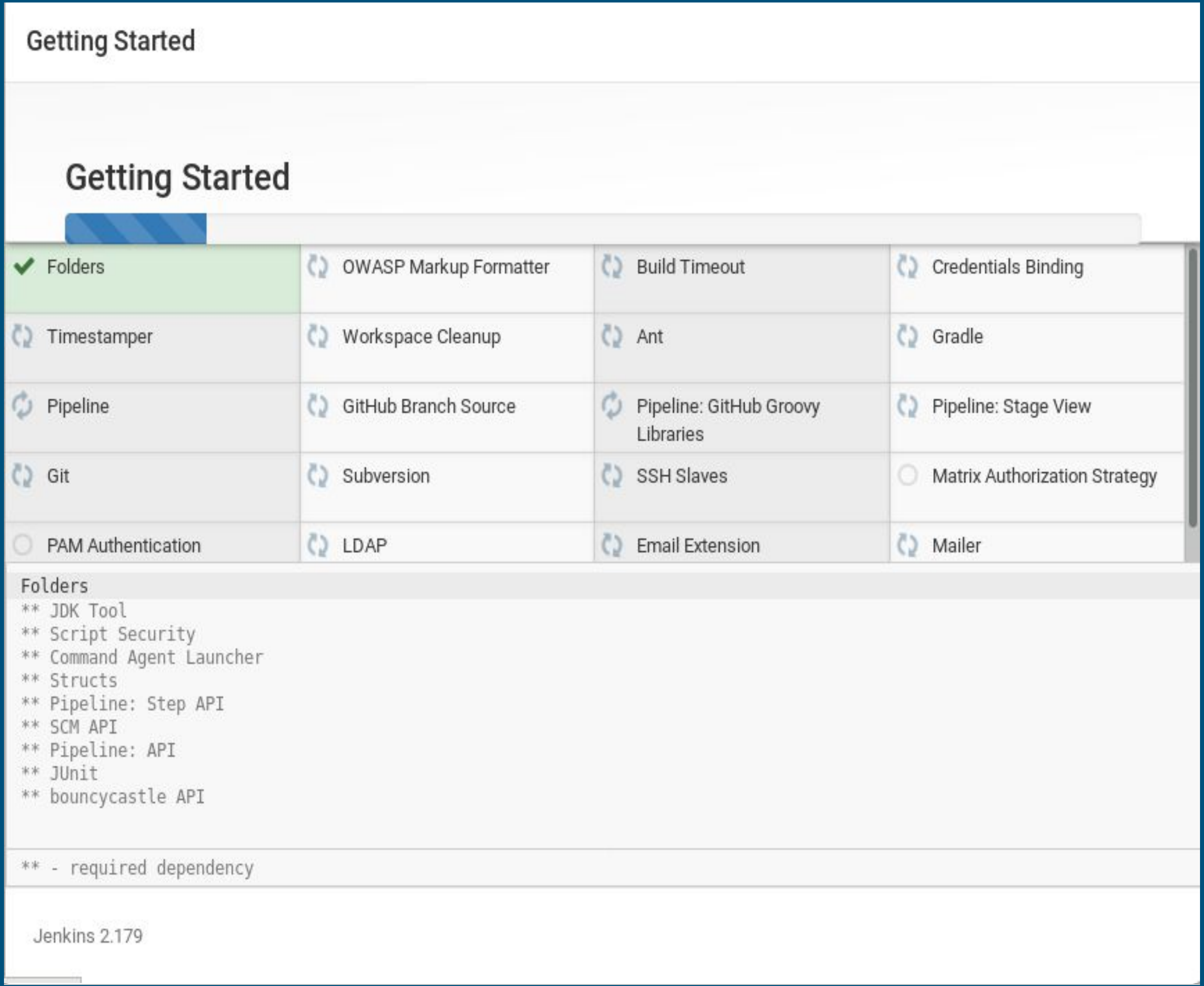
Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.179

Setup

After selecting "Install suggested plugins":



Getting Started

Getting Started

✓ Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding
Timestamper	Workspace Cleanup	Ant	Gradle
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View
Git	Subversion	SSH Slaves	Matrix Authorization Strategy
PAM Authentication	LDAP	Email Extension	Mailer

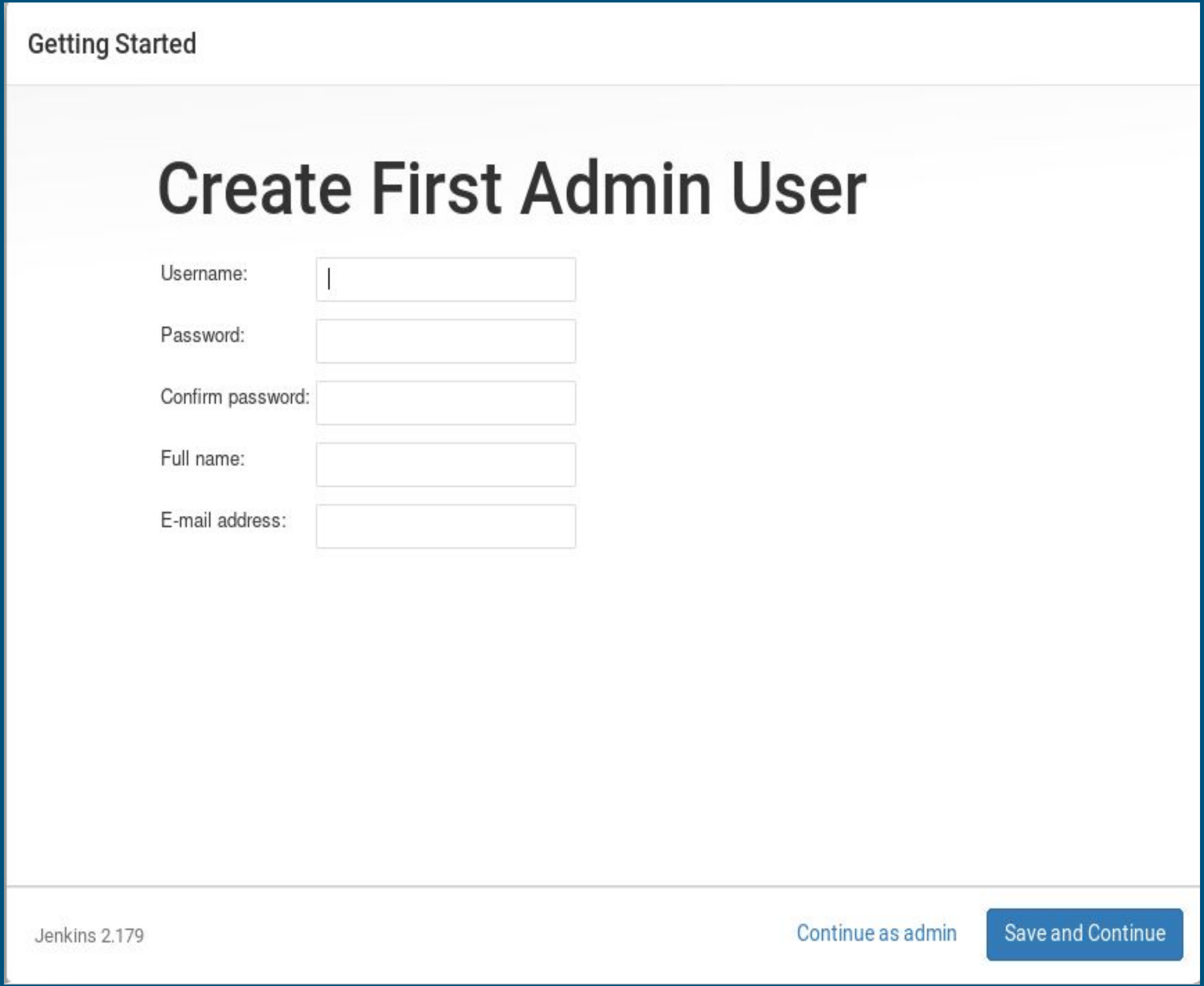
```
Folders
** JDK Tool
** Script Security
** Command Agent Launcher
** Structs
** Pipeline: Step API
** SCM API
** Pipeline: API
** JUnit
** bouncycastle API
```

```
** - required dependency
```

Jenkins 2.179

Setup

After “Install suggested plugins” have completed, you will be brought to this page automatically :


Getting Started

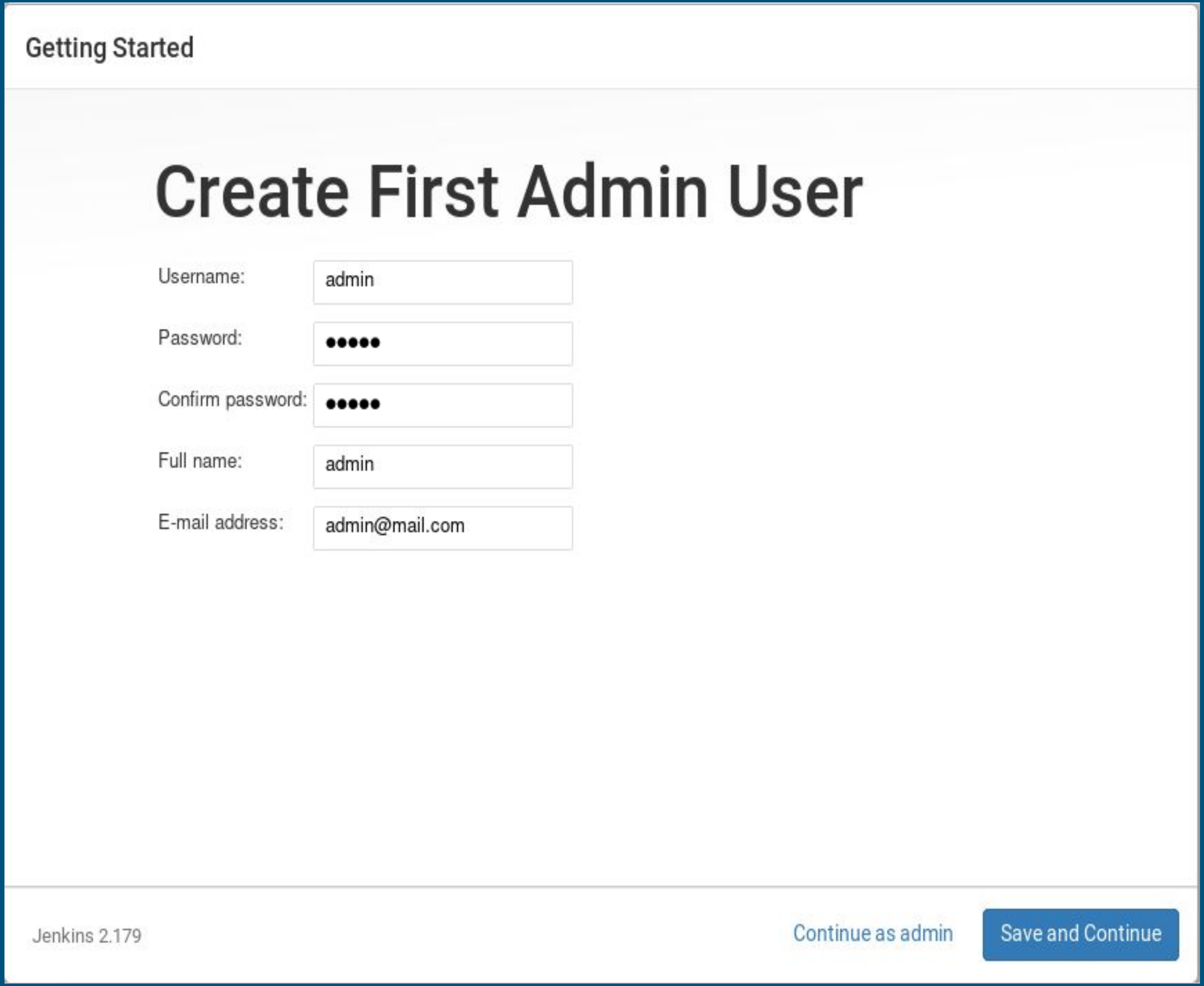
Create First Admin User

Username:	
Password:	
Confirm password:	
Full name:	
E-mail address:	

Setup

Create an admin username and password of your choice.

Jenkins will use the provided email address to send notifications (if configured in settings - can be changed later)



Getting Started

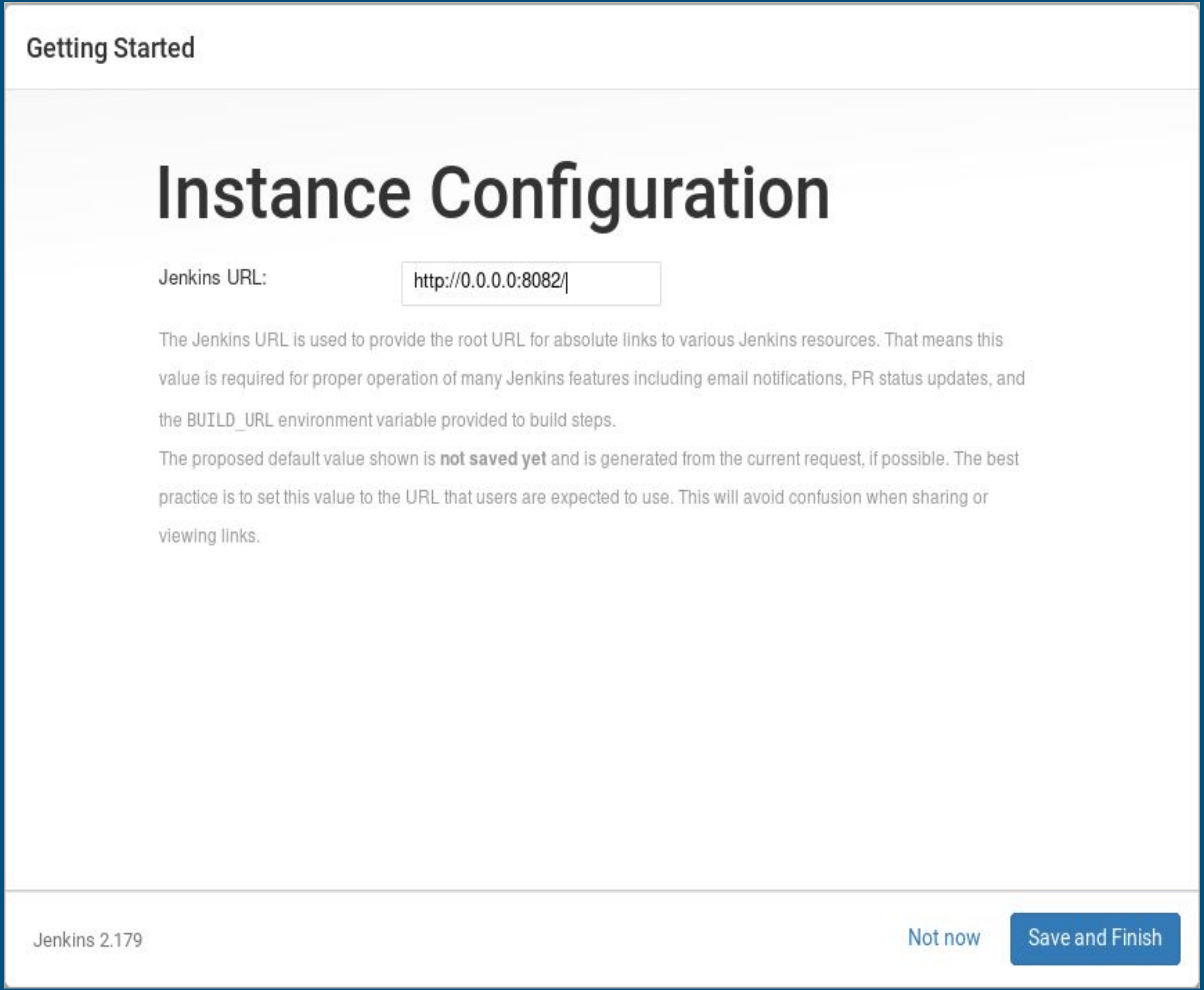
Create First Admin User

Username:	admin
Password:	•••••
Confirm password:	•••••
Full name:	admin
E-mail address:	admin@mail.com

Setup

This is running on a personal computer (it is not recommended see slide 4)

The “Jenkins URL” field should be automatically populated with the IP of where Jenkins is running.



Getting Started

Instance Configuration

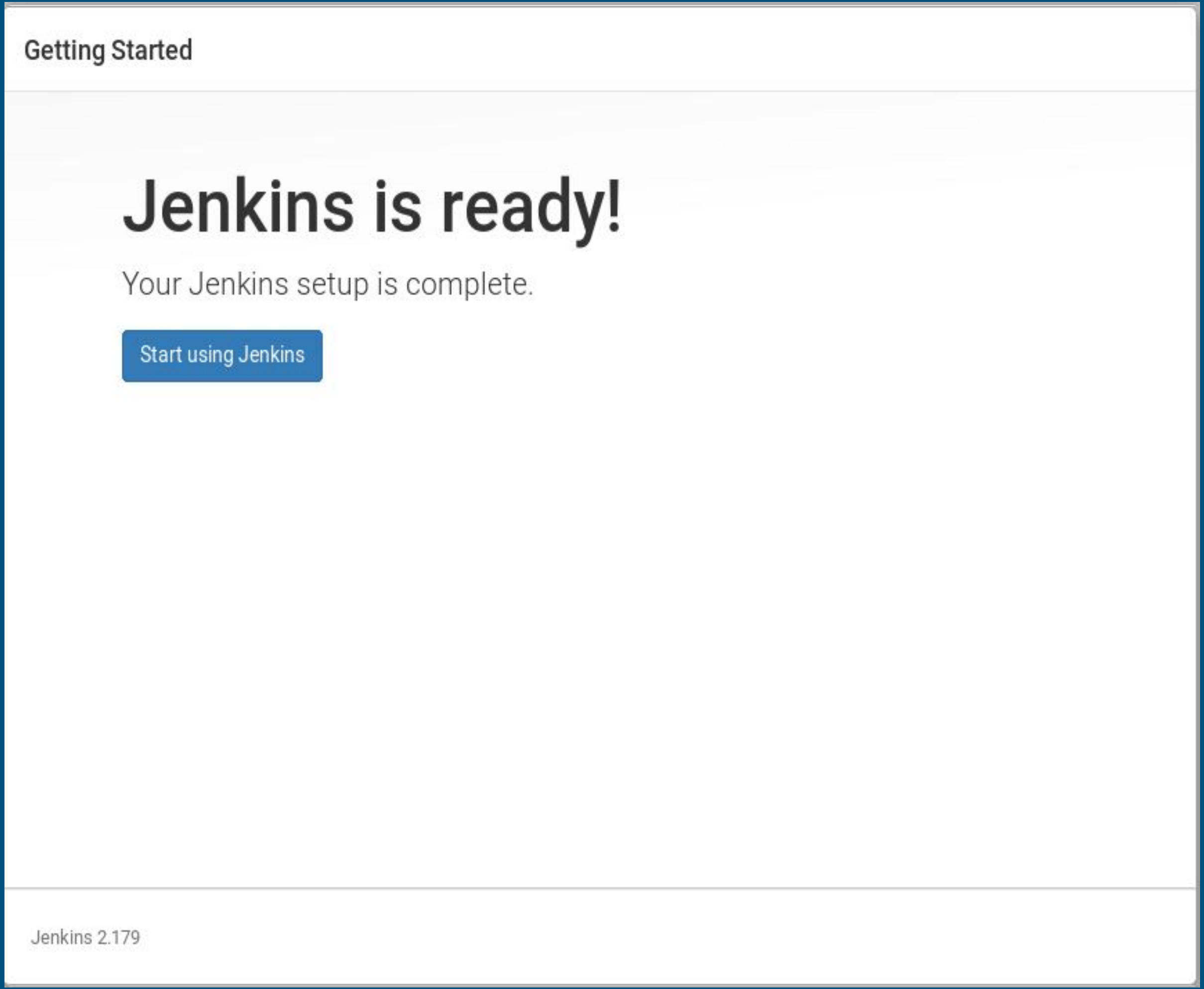
Jenkins URL: http://0.0.0.0:8082/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Setup

After clicking “Save and Finish” , click “Start using Jenkins”.

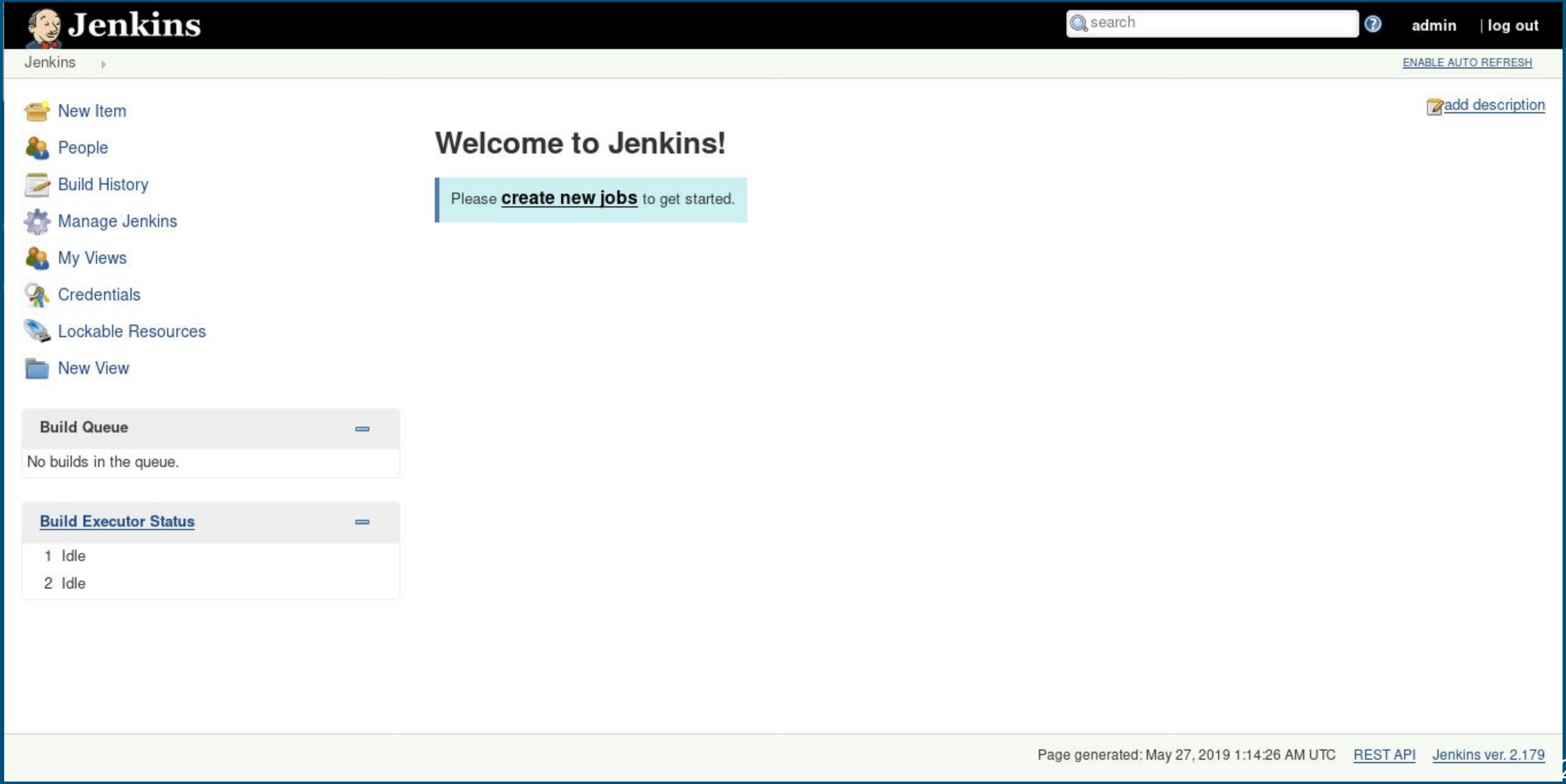

Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

Jenkins 2.179

Setup Complete


Jenkins

search

admin | log out

Jenkins

ENABLE AUTO REFRESH

add description

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Credentials
- Lockable Resources
- New View

Welcome to Jenkins!

Please **create new jobs** to get started.

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

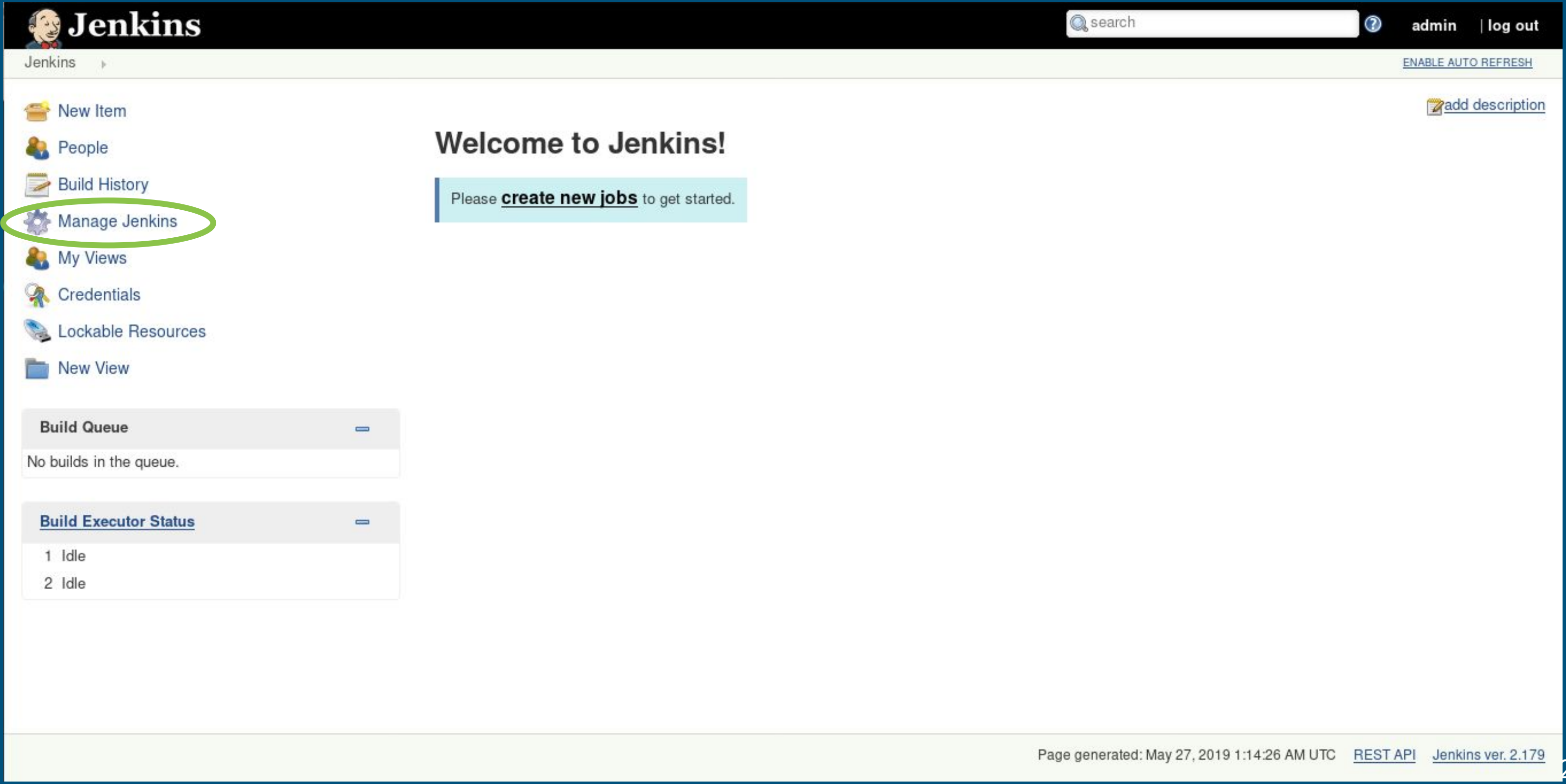
Page generated: May 27, 2019 1:14:26 AM UTC REST API Jenkins ver. 2.179

Connect Jenkins to GitHub

This “Connect Jenkins to GitHub” section will guide you through setting up Jenkins to pull code from a GitHub repository via the the Jenkins GitHub Plugin.

Click “Manage Jenkins” on the left side panel of the dashboard to get started.

Connect Jenkins to GitHub



Welcome to Jenkins!

Please **create new jobs** to get started.

Build Queue

No builds in the queue.

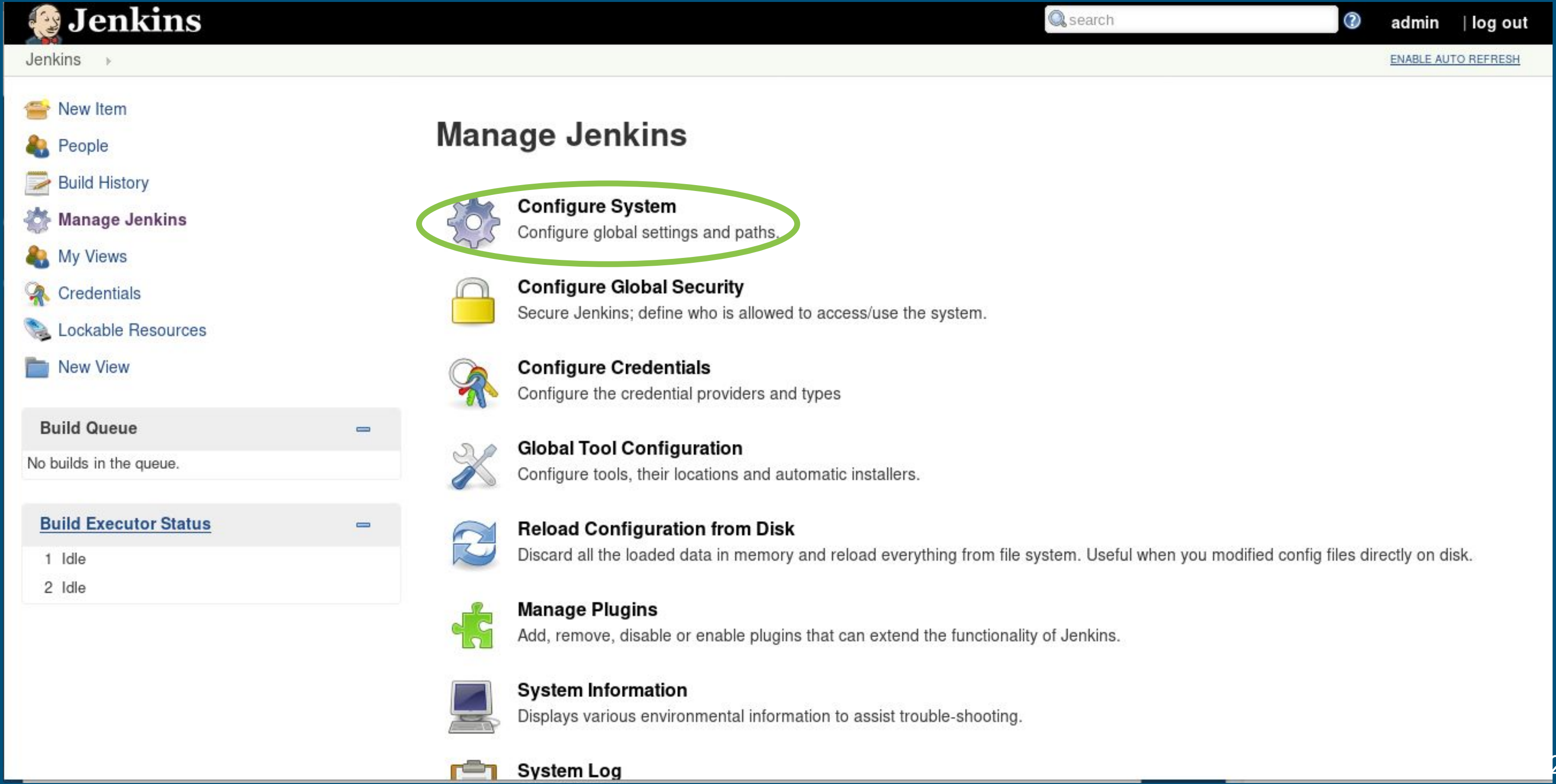
Build Executor Status

1 Idle

2 Idle

Page generated: May 27, 2019 1:14:26 AM UTC REST API Jenkins ver. 2.179

Connect Jenkins to GitHub



Jenkins | search | admin | log out

Jenkins ▸ ENABLE AUTO REFRESH

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Credentials
- Lockable Resources
- New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Manage Jenkins

Configure System
Configure global settings and paths.

Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.

Configure Credentials
Configure the credential providers and types

Global Tool Configuration
Configure tools, their locations and automatic installers.

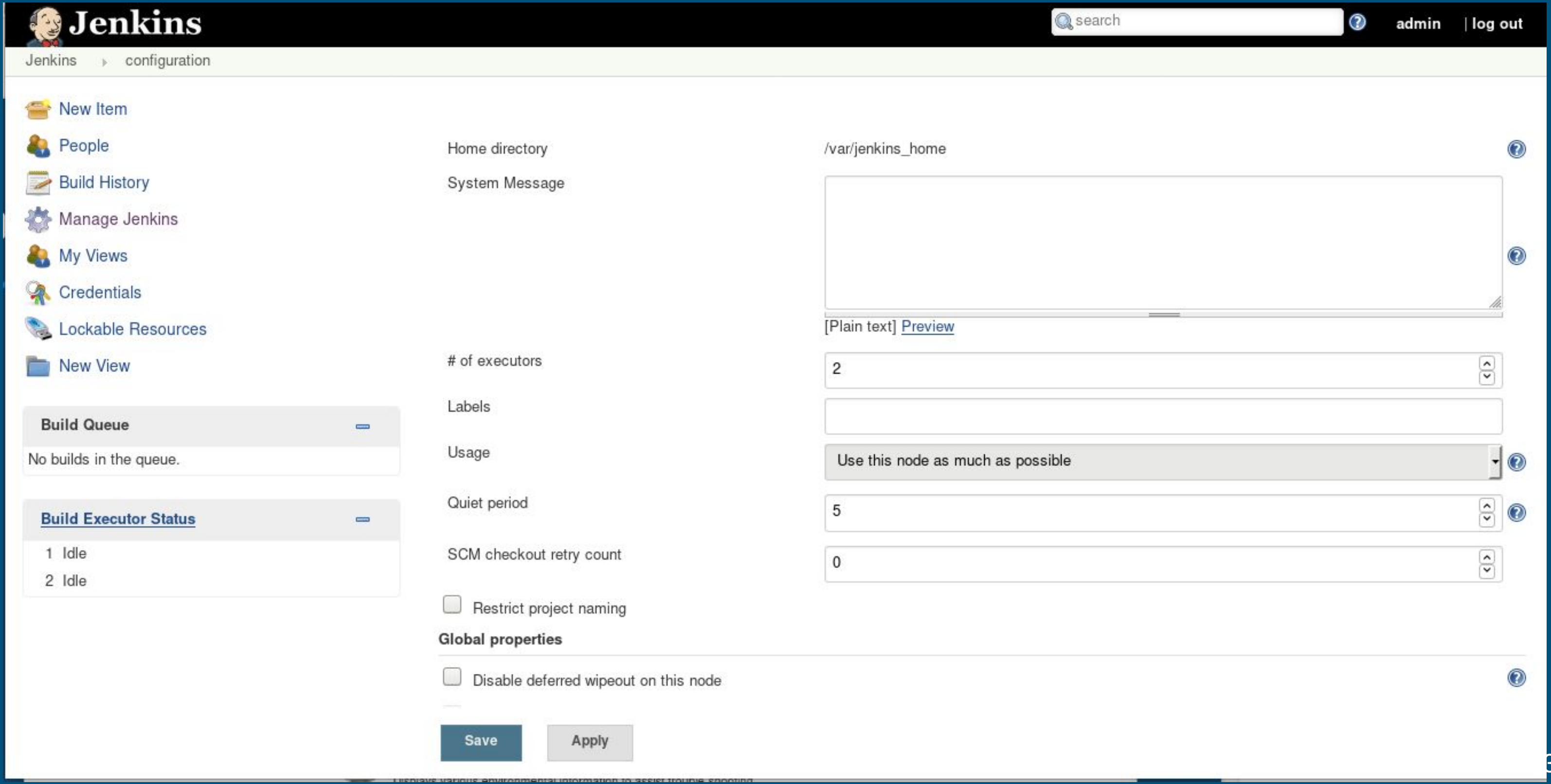
Reload Configuration from Disk
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.

Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

System Information
Displays various environmental information to assist trouble-shooting.

System Log

Connect Jenkins to GitHub



No builds in the queue.

1 Idle
2 Idle

Home directory /var/jenkins_home

System Message

[Plain text] Preview

# of executors: 2

Labels:

Usage: Use this node as much as possible

Quiet period: 5

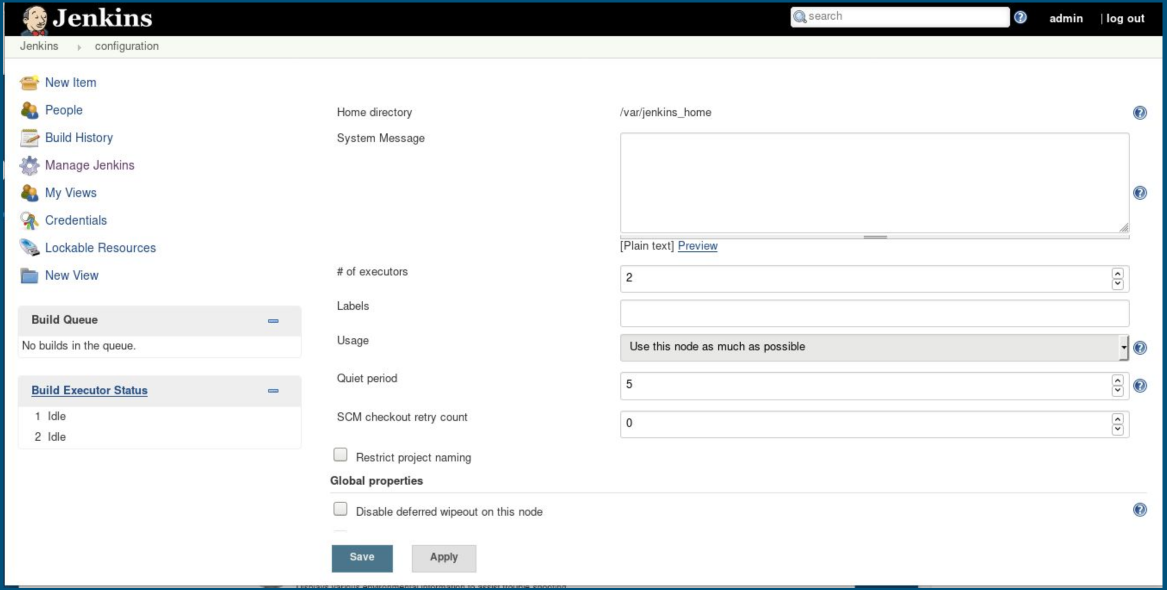
SCM checkout retry count: 0

☐ Restrict project naming

Global properties

☐ Disable deferred wipeout on this node

Connect Jenkins to GitHub



No builds in the queue.

1 Idle
2 Idle

Home directory	/var/jenkins_home
System Message	
	[Plain text] Preview
# of executors	2
Labels	
Usage	Use this node as much as possible
Quiet period	5
SCM checkout retry count	0

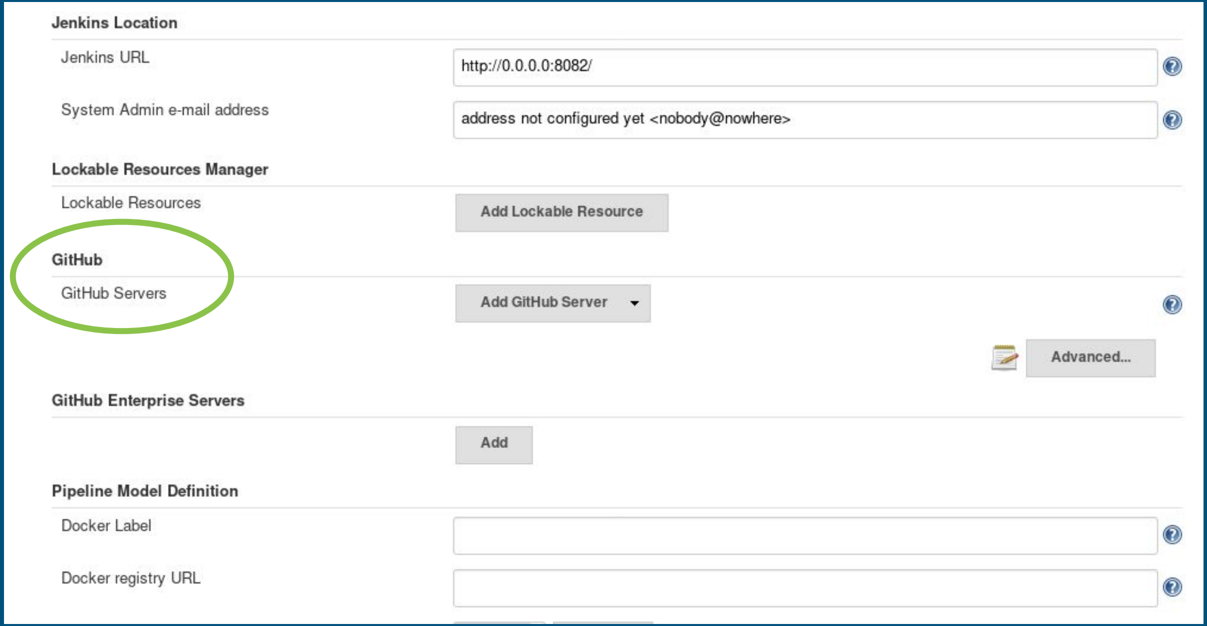
☐ Restrict project naming

Global properties

☐ Disable deferred wipeout on this node

Scroll down to locate the GitHub section.

Connect Jenkins to GitHub



Jenkins Location

Jenkins URL	http://0.0.0.0:8082/
System Admin e-mail address	address not configured yet <nobody@nowhere>

Lockable Resources Manager

Lockable Resources

GitHub

GitHub Servers

GitHub Enterprise Servers

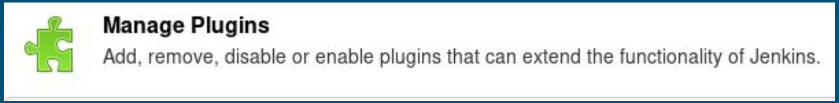
Pipeline Model Definition

Docker Label	
Docker registry URL	

This GitHub section may appear in different parts of "Configure System" depending on the Jenkins version. Version shown on slidedeck *`2.179`*

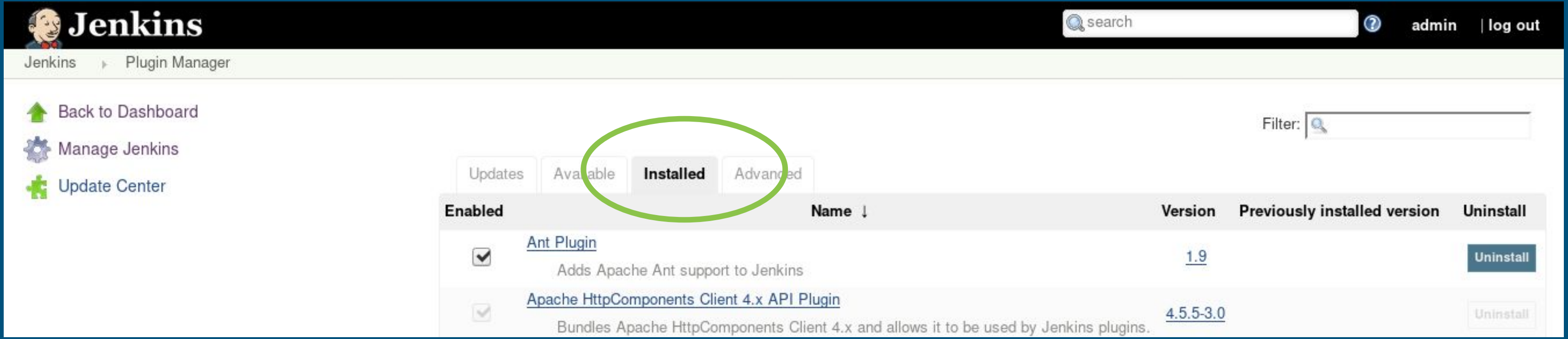
Connect Jenkins to GitHub

If you cannot find the GitHub section on “Configure system”. Select “Manage Plugins” from “Manage Jenkins” (found on the left side panel of the dashboard).



Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Select “Installed” and scroll down to look for the GitHub Plugin.



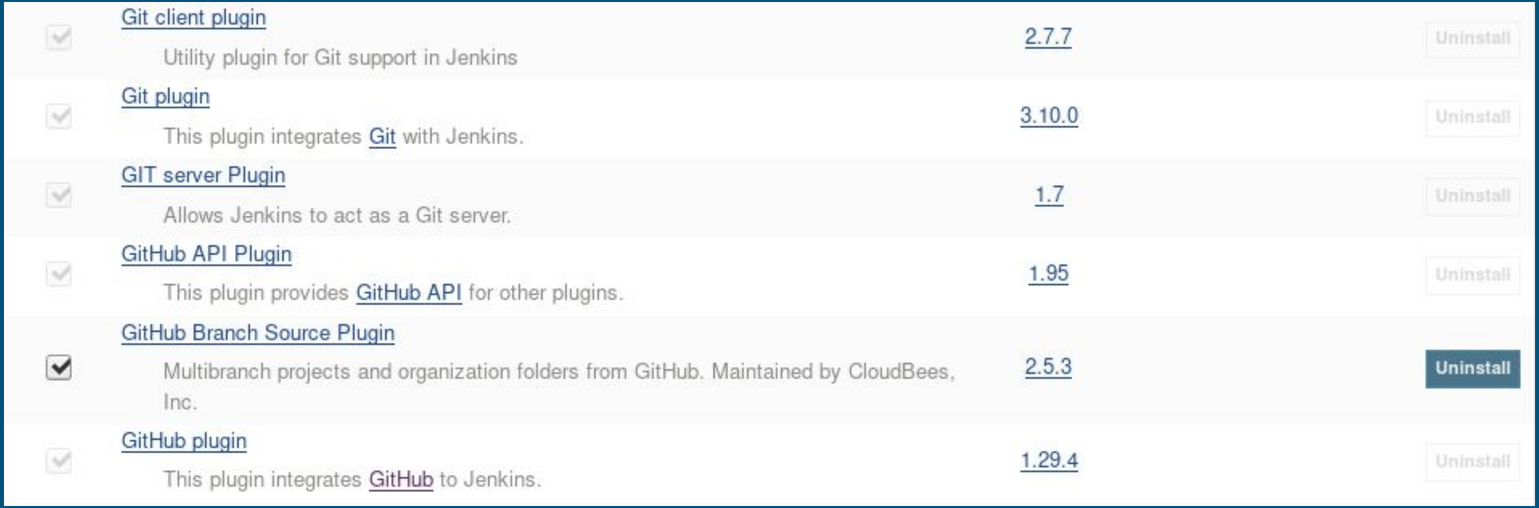
Jenkins › Plugin Manager

- Back to Dashboard
- Manage Jenkins
- Update Center

Enabled	Name ↓	Version	Previously installed version	Uninstall
✓	Ant Plugin Adds Apache Ant support to Jenkins	1.9		Uninstall
✓	Apache HttpComponents Client 4.x API Plugin Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins.	4.5.5-3.0		Uninstall

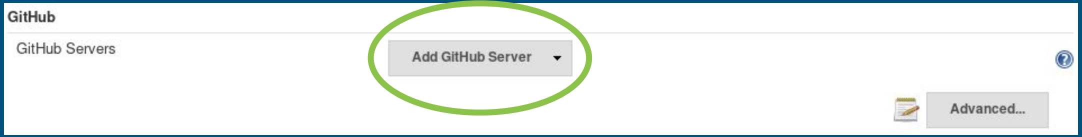
Connect Jenkins to GitHub

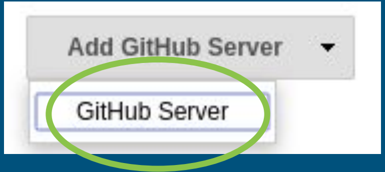
If you cannot find the GitHub section on “Configure system”. Make sure you have the following plugins installed. Especially the “GitHub plugin”.



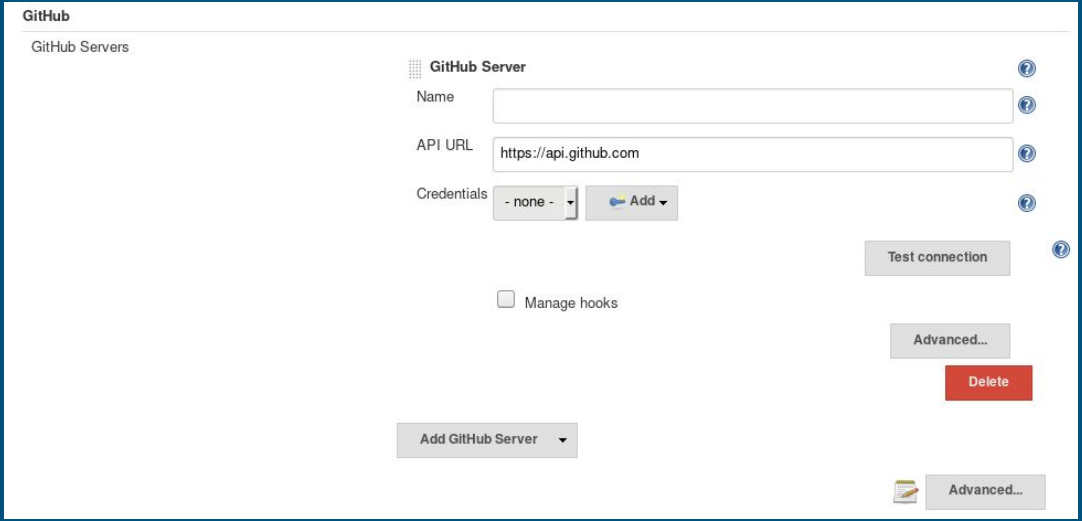
✓	Git client plugin Utility plugin for Git support in Jenkins	2.7.7	Uninstall
✓	Git plugin This plugin integrates Git with Jenkins.	3.10.0	Uninstall
✓	GIT server Plugin Allows Jenkins to act as a Git server.	1.7	Uninstall
✓	GitHub API Plugin This plugin provides GitHub API for other plugins.	1.95	Uninstall
✓	GitHub Branch Source Plugin Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.	2.5.3	Uninstall
✓	GitHub plugin This plugin integrates GitHub to Jenkins.	1.29.4	Uninstall

Connect Jenkins to GitHub





Select “Add GitHub Server”

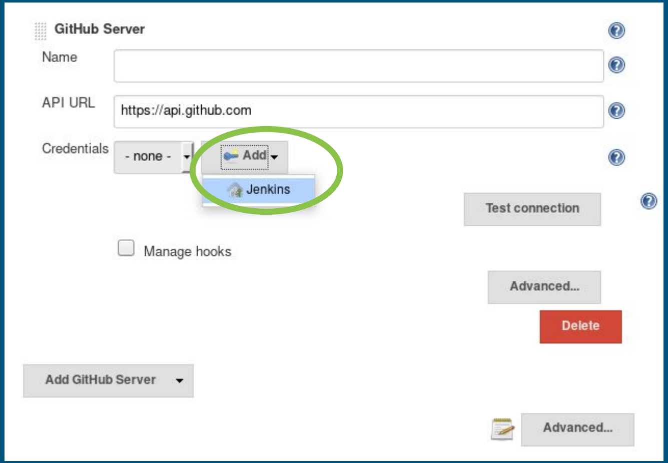


Name
API URL https://api.github.com
Credentials - none -

Leave the “API URL” as is.

Provide any name of your choice to the “Name” field.

Connect Jenkins to GitHub

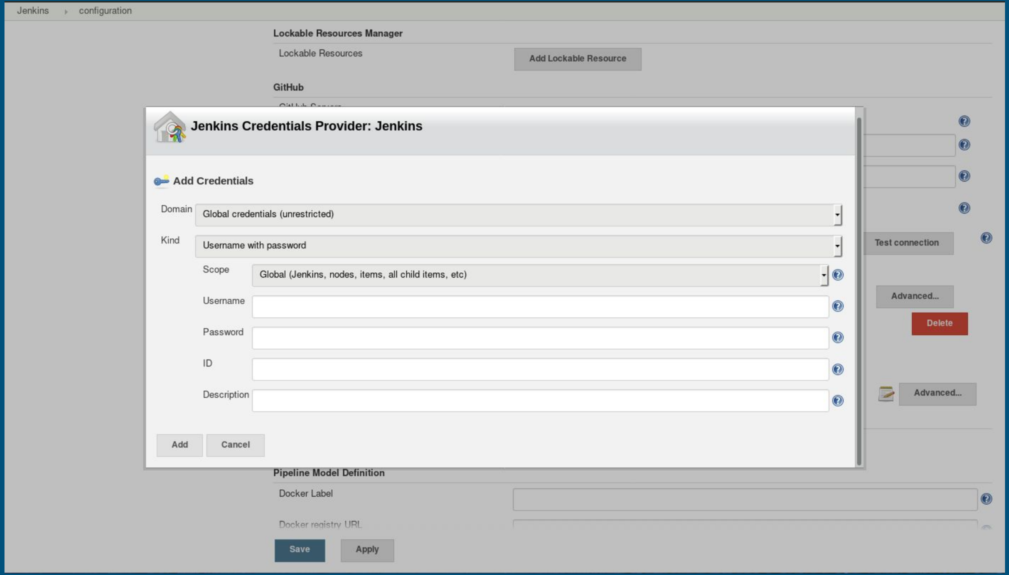


GitHub Server

Name:

API URL: https://api.github.com

Credentials: - none -



Jenkins › configuration

Lockable Resources Manager

Lockable Resources

GitHub

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

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

Username:

Password:

ID:

Description:

Pipeline Model Definition

Docker Label

Docker registry URL

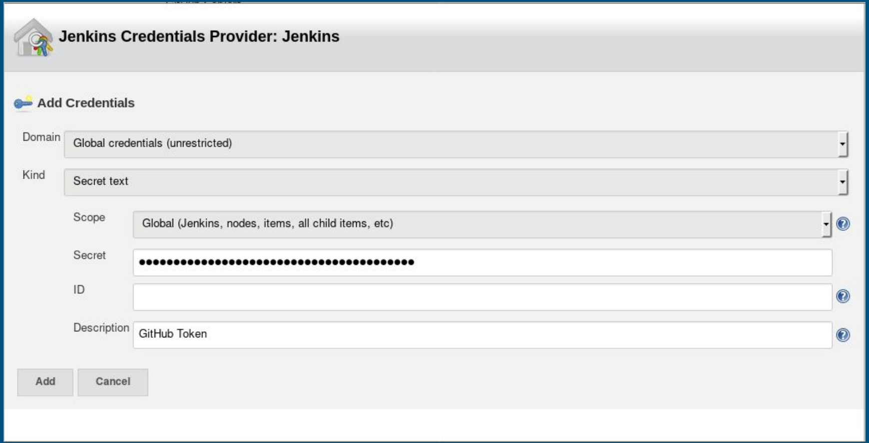
Add a new Jenkins Credential.

Connect Jenkins to GitHub

Have the following fields filled as shown in the screenshot below:

The “Description” can be anything of your choice to describe the Secret Text.

The “ID” field can be left blank.



Jenkins Credentials Provider: Jenkins

Add Credentials

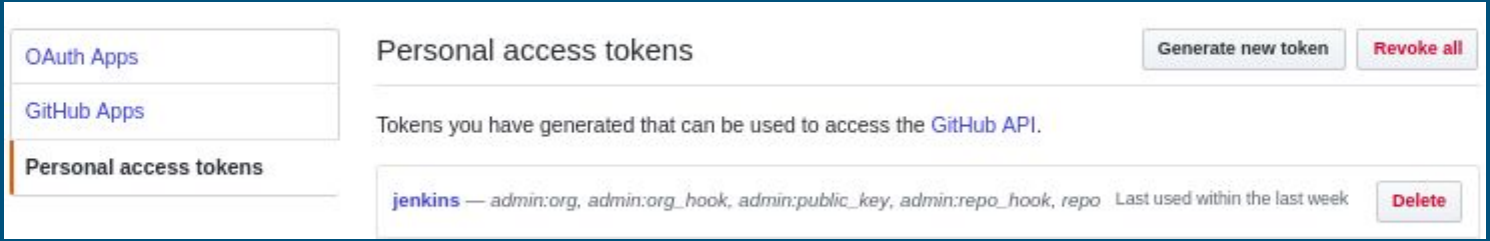
Domain	Global credentials (unrestricted)
Kind	Secret text
Scope	Global (Jenkins, nodes, items, all child items, etc)
Secret	●●
ID	
Description	GitHub Token

To generate the “Secret” i.e. a token generated by GitHub see the next slide.

Connect Jenkins to GitHub

Log into GitHub and go to https://github.com/settings/tokens . Use the GitHub account that has access to the repository that you want Jenkins to pull code from.

Select “Personal access tokens” and click on “Generate new token”:



Personal access tokens

Tokens you have generated that can be used to access the GitHub API.

jenkins — *admin:org, admin:org_hook, admin:public_key, admin:repo_hook, repo* Last used within the last week

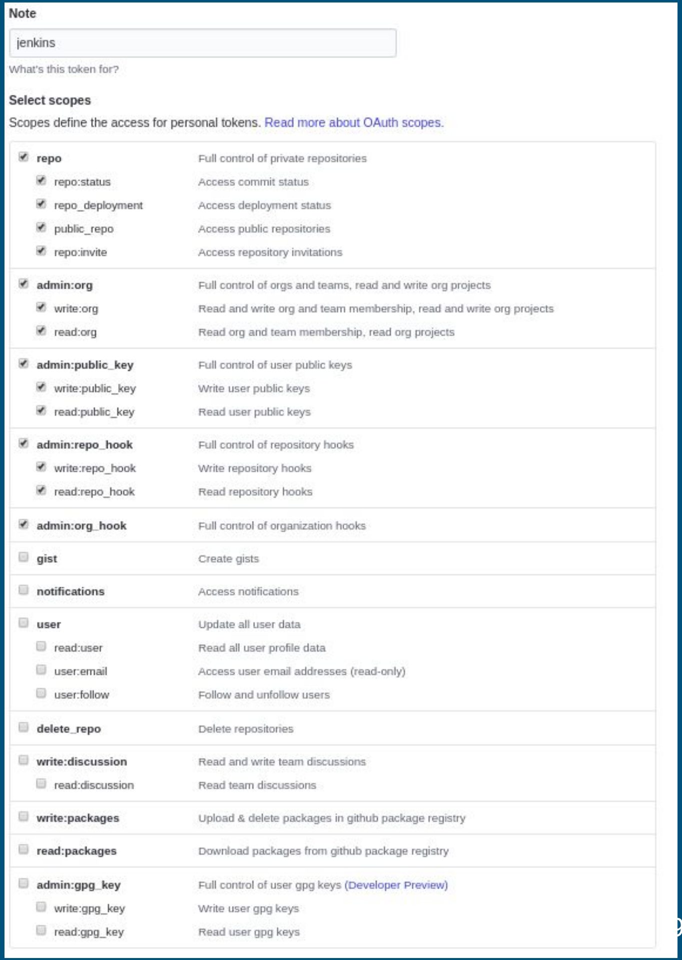
Connect Jenkins to GitHub

Select the follow scopes as shown in the screenshot.

If your repository is part of a organization, be sure to select “admin:org” and “admin:org_hook”.

Provide a descriptive note for this token in the “Note” field.

When done select, Generate token .



Note

jenkins

What's this token for?

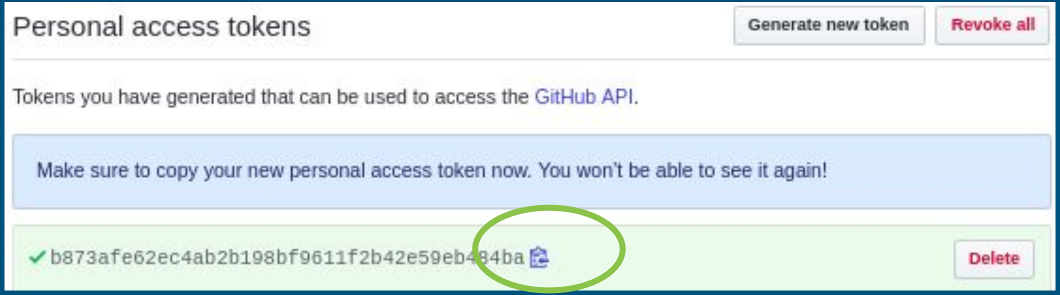
Select scopes

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

✓	**repo**	Full control of private repositories
✓	repo:status	Access commit status
✓	repo_deployment	Access deployment status
✓	public_repo	Access public repositories
✓	repo:invite	Access repository invitations
✓	**admin:org**	Full control of orgs and teams, read and write org projects
✓	write:org	Read and write org and team membership, read and write org projects
✓	read:org	Read org and team membership, read org projects
✓	**admin:public_key**	Full control of user public keys
✓	write:public_key	Write user public keys
✓	read:public_key	Read user public keys
✓	**admin:repo_hook**	Full control of repository hooks
✓	write:repo_hook	Write repository hooks
✓	read:repo_hook	Read repository hooks
✓	**admin:org_hook**	Full control of organization hooks
☐	**gist**	Create gists
☐	**notifications**	Access notifications
☐	**user**	Update all user data
☐	read:user	Read all user profile data
☐	user:email	Access user email addresses (read-only)
☐	user:follow	Follow and unfollow users
☐	**delete_repo**	Delete repositories
☐	**write:discussion**	Read and write team discussions
☐	read:discussion	Read team discussions
☐	**write:packages**	Upload & delete packages in github package registry
☐	**read:packages**	Download packages from github package registry
☐	**admin:gpg_key**	Full control of user gpg keys (Developer Preview)
☐	write:gpg_key	Write user gpg keys
☐	read:gpg_key	Read user gpg keys

Connect Jenkins to GitHub

Copy the hash as shown in the screenshot, you can click on this icon to easily copy it.



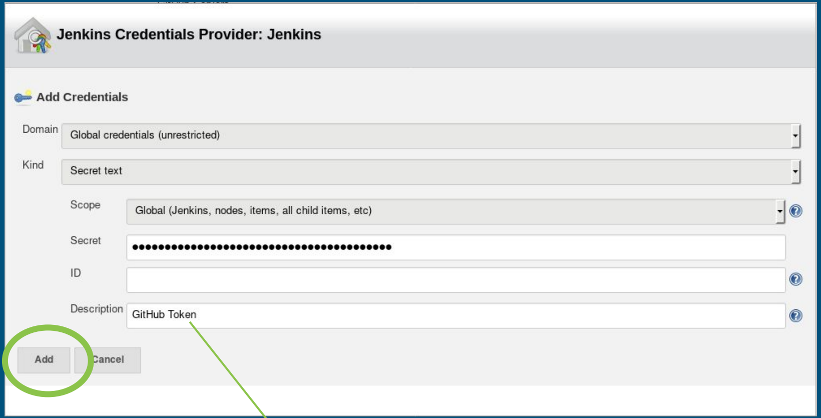
Personal access tokens

Tokens you have generated that can be used to access the GitHub API.

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

✓ b873afe62ec4ab2b198bf9611f2b42e59eb484ba

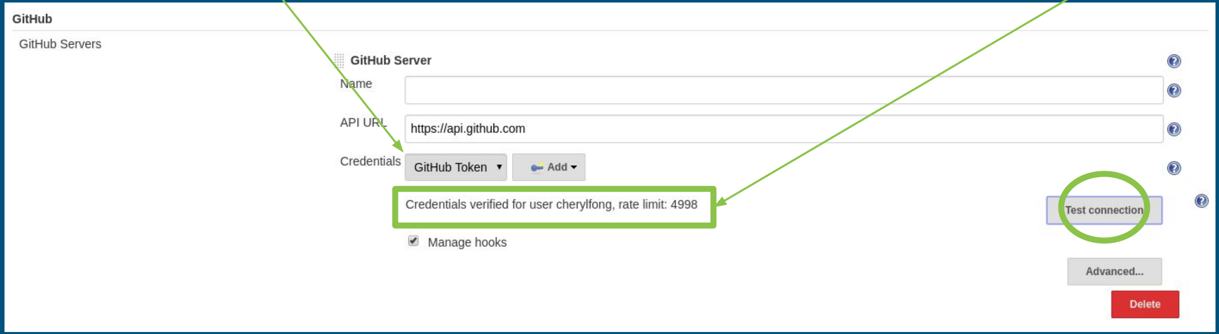
Connect Jenkins to GitHub



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain	Global credentials (unrestricted)
Kind	Secret text
Scope	Global (Jenkins, nodes, items, all child items, etc)
Secret	••
ID	
Description	GitHub Token



GitHub

GitHub Servers

GitHub Server

Name	
API URL	https://api.github.com
Credentials	GitHub Token

Credentials verified for user cherylfong, rate limit: 4998

Manage hooks ✓

After clicking "Add", test the connection.

If successful, it should state your GitHub username and the rate limit.

Connect Jenkins to GitHub

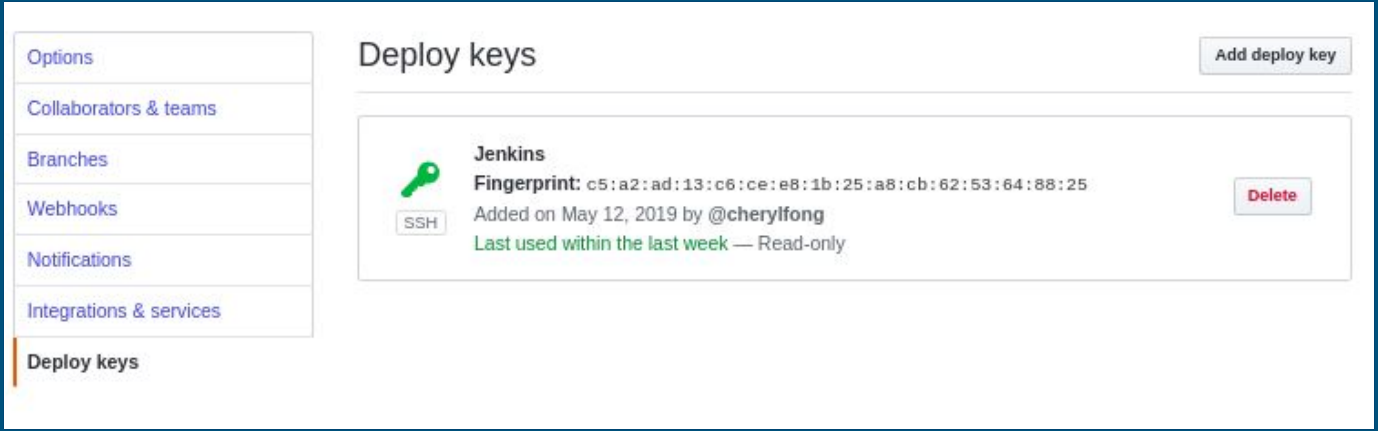
Up to this point, Jenkins will be able to access the source code on a repository via the GitHub Token. You need to make sure that the Token was created through an account that has push/pull access to the repository.

The settings on the next slides will allow Jenkins to push build status updates on GitHub and be triggered for builds when commits are pushed onto the “master” branch.

Connect Jenkins to GitHub

Set up a “Deploy key” on your repository, by going to:

https://github.com/<name-of-organization>/<name-of-repository>/settings/keys

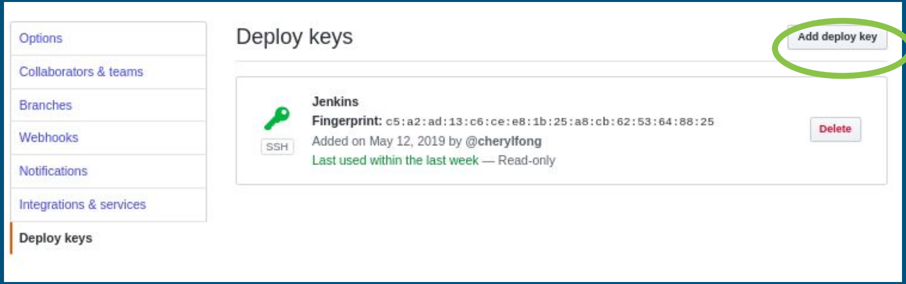


Deploy keys

Jenkins
Fingerprint: c5:a2:ad:13:c6:ce:e8:1b:25:a8:cb:62:53:64:88:25
Added on May 12, 2019 by @cherylfong
Last used within the last week — Read-only

Connect Jenkins to GitHub

Select “Add deploy key”

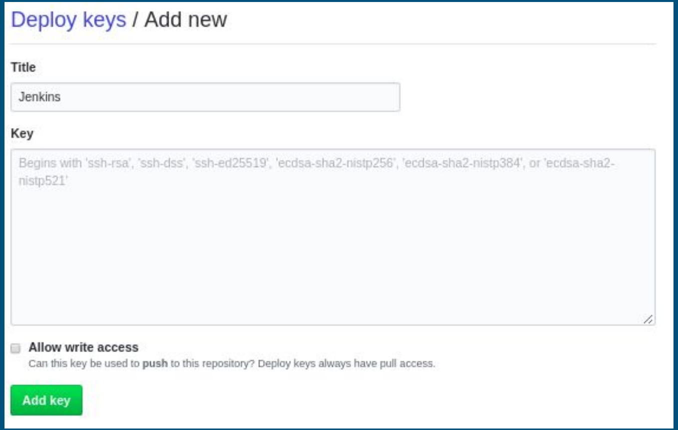


Deploy keys

Add deploy key

Jenkins
Fingerprint: `c5:a2:ad:13:c6:ce:e8:1b:25:a8:cb:62:53:64:88:25`
SSH
Added on May 12, 2019 by @cherylfong
Last used within the last week — Read-only

Delete



Deploy keys / Add new

Title

Jenkins

Key

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

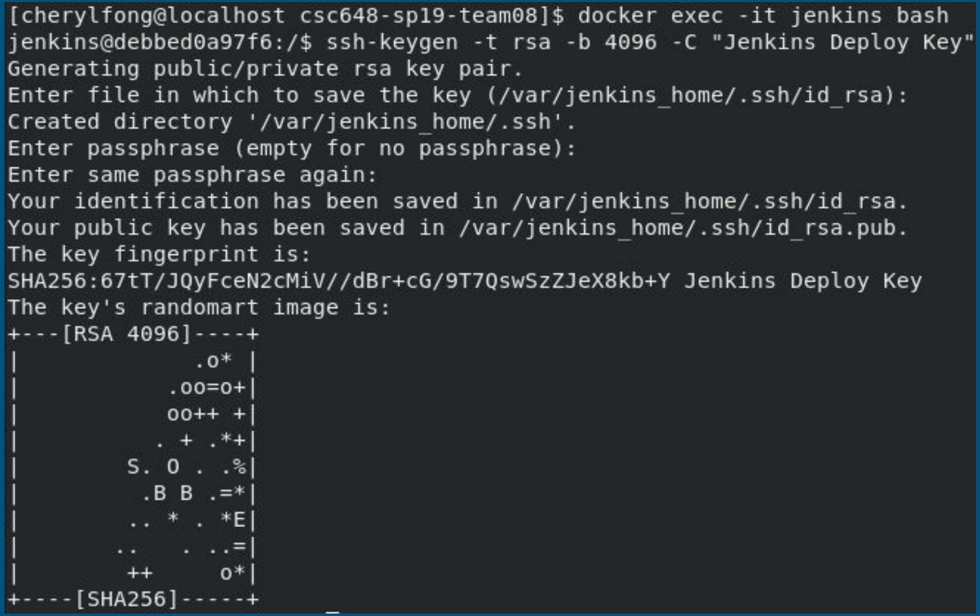
☐ **Allow write access**
Can this key be used to **push** to this repository? Deploy keys always have pull access.

Add key

See this GitHub guide on Generating a new SSH Key and adding it to the SSH Agent (Official GitHub Guide).

Connect Jenkins to GitHub

To Generate an SSH key on the Jenkins Image container, follow the commands in this screenshot:



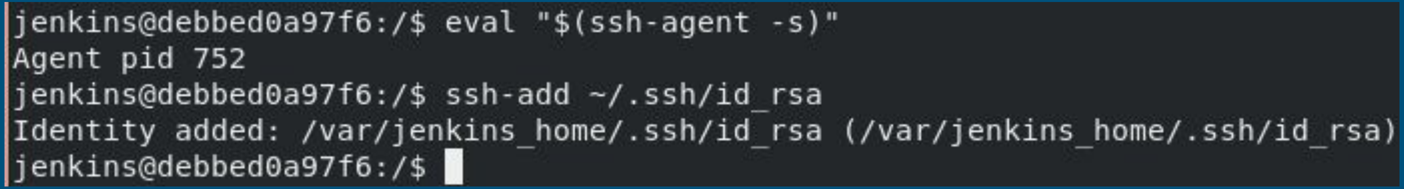
```
[cherylfong@localhost csc648-sp19-team08]$ docker exec -it jenkins bash
jenkins@debbed0a97f6:/$ ssh-keygen -t rsa -b 4096 -C "Jenkins Deploy Key"
Generating public/private rsa key pair.
Enter file in which to save the key (/var/jenkins_home/.ssh/id_rsa):
Created directory '/var/jenkins_home/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/jenkins_home/.ssh/id_rsa.
Your public key has been saved in /var/jenkins_home/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:67tT/JQyFceN2cMiV//dBr+cG/9T7QswSzZJeX8kb+Y Jenkins Deploy Key
The key's randomart image is:
+---[RSA 4096]----+
|             .o* |
|           .oo=o+|
|           oo++ +|
|          . + .*+|
|        S. O . .%|
|         .B B .=*|
|        .. * . *E|
|       ..   . ..=|
|        ++      o*|
+----[SHA256]-----+
```

The passphrase was left blank deliberately.

Note down the passphrase if you choose to have it for this SSH key.

Connect Jenkins to GitHub

Finally add the SSH key to the SSH Agent.

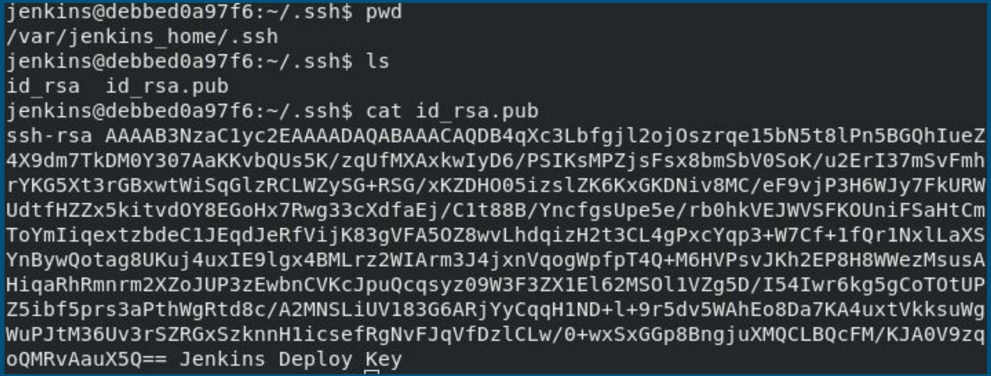


```
jenkins@debbed0a97f6:/$ eval "$(ssh-agent -s)"
Agent pid 752
jenkins@debbed0a97f6:/$ ssh-add ~/.ssh/id_rsa
Identity added: /var/jenkins_home/.ssh/id_rsa (/var/jenkins_home/.ssh/id_rsa)
jenkins@debbed0a97f6:/$ 
```

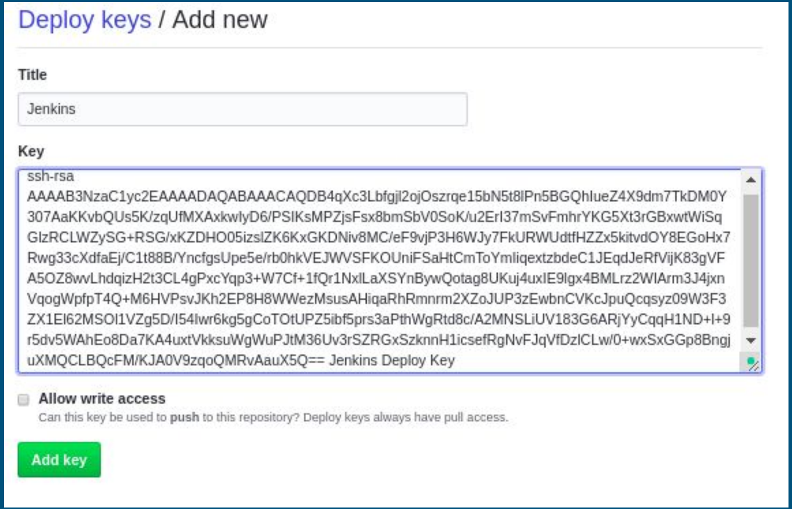
Slide 45 and 46 utilized content from Generating a new SSH Key and adding it to the SSH Agent (Official GitHub Guide).

Connect Jenkins to GitHub

After generating the SSH key pair, copy and paste the contents of the public key and paste it into the Key filed.



```
jenkins@debbed0a97f6:~/.ssh$ pwd
/var/jenkins_home/.ssh
jenkins@debbed0a97f6:~/.ssh$ ls
id_rsa  id_rsa.pub
jenkins@debbed0a97f6:~/.ssh$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDB4qXc3Lbfgjl2ojOszrqe15bN5t8lPn5BGQhIueZ
4X9dm7TkDM0Y307AaKKvbQUs5K/zqUfMXAxkwIyD6/PSIKsMPZjsFsx8bmSbV0SoK/u2ErI37mSvFmh
rYKG5Xt3rGBxwtWiSqGlzRCLWZySG+RSG/xKZDHO05izslZK6KxGKDNiv8MC/eF9vjP3H6WJy7FkURW
UdtfHZZx5kitvdOY8EGoHx7Rwg33cXdfaEj/C1t88B/YncfgsUpe5e/rb0hkVEJWVSFKOUniFSaHtCm
ToYmIiqextzbdeC1JEqdJeRfVijK83gVFA5OZ8wvLhdqizH2t3CL4gPxcYqp3+W7Cf+1fQr1NxlLaXS
YnBywQotag8UKuj4uxIE9lgx4BMLrz2WIArm3J4jxnVqogWpfpT4Q+M6HVPsvJKh2EP8H8WWezMsusA
HiqaRhRmnrm2XZoJUP3zEwbnCVKcJpuQcqsyz09W3F3ZX1El62MSOl1VZg5D/I54Iwr6kg5gCoTOtUP
Z5ibf5prs3aPthWgRtd8c/A2MNSLiUV183G6ARjYyCqqH1ND+l+9r5dv5WAhEo8Da7KA4uxtVkksuWg
WuPJtM36Uv3rSZRGxSzknnH1icsefRgNvFJqVfDzlCLw/0+wxSxGGp8BngjuXMQCLBQcFM/KJA0V9zq
oQMRvAauX5Q== Jenkins Deploy Key
```



Deploy keys / Add new

Title

Jenkins

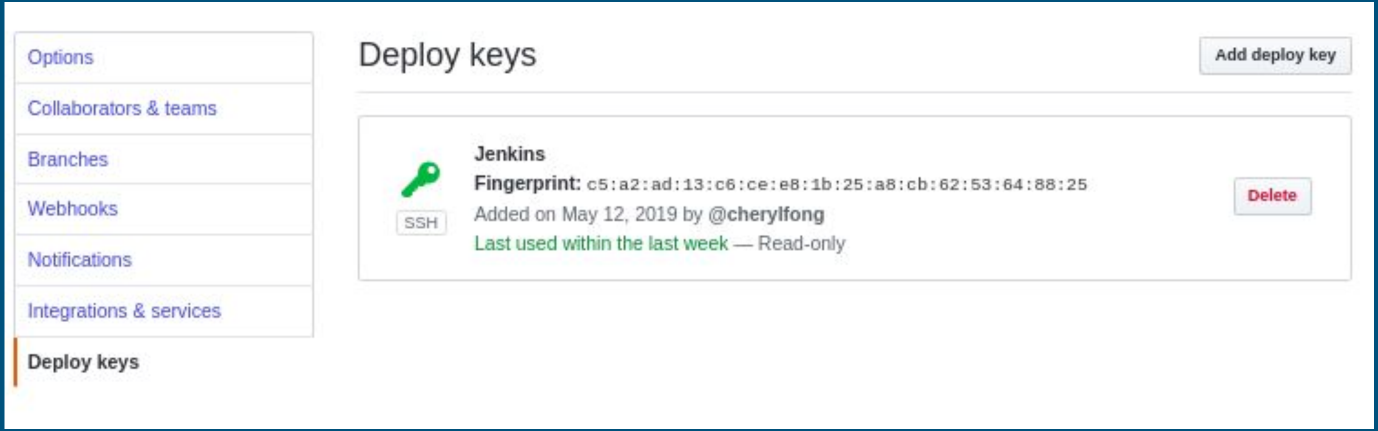
Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDB4qXc3Lbfgjl2ojOszrqe15bN5t8lPn5BGQhIueZ4X9dm7TkDM0Y
307AaKKvbQUs5K/zqUfMXAxkwIyD6/PSIKsMPZjsFsx8bmSbV0SoK/u2ErI37mSvFmhrYKG5Xt3rGBxwtWiSq
GlzRCLWZySG+RSG/xKZDHO05izslZK6KxGKDNiv8MC/eF9vjP3H6WJy7FkURWUdtfHZZx5kitvdOY8EGoHx7
Rwg33cXdfaEj/C1t88B/YncfgsUpe5e/rb0hkVEJWVSFKOUniFSaHtCmToYmIiqextzbdeC1JEqdJeRfVijK83gVF
A5OZ8wvLhdqizH2t3CL4gPxcYqp3+W7Cf+1fQr1NxlLaXSYnBywQotag8UKuj4uxIE9lgx4BMLrz2WIArm3J4jxn
VqogWpfpT4Q+M6HVPsvJKh2EP8H8WWezMsusAHiqaRhRmnrm2XZoJUP3zEwbnCVKcJpuQcqsyz09W3F3
ZX1El62MSOl1VZg5D/I54Iwr6kg5gCoTOtUPZ5ibf5prs3aPthWgRtd8c/A2MNSLiUV183G6ARjYyCqqH1ND+l+9
r5dv5WAhEo8Da7KA4uxtVkksuWgWuPJtM36Uv3rSZRGxSzknnH1icsefRgNvFJqVfDzlCLw/0+wxSxGGp8Bngj
uXMQCLBQcFM/KJA0V9zqoQMRvAauX5Q== Jenkins Deploy Key
```

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

Leave the "Allow write access" unchecked and click "Add key".

Connect Jenkins to GitHub



Deploy keys

Jenkins
Fingerprint: `c5:a2:ad:13:c6:ce:e8:1b:25:a8:cb:62:53:64:88:25`
Added on May 12, 2019 by **@cherylfong**
Last used within the last week — Read-only

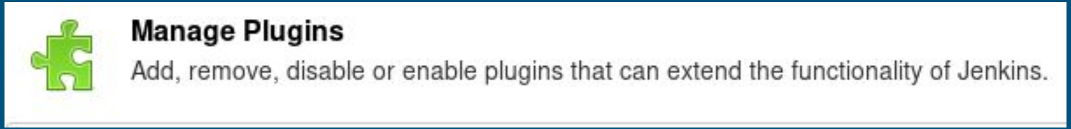
At this point you should have the above. See slide 64 on the Deployment Example section to see how this will be used.

SSH via Jenkins

This section “SSH” via Jenkins, will provide Jenkins permissions to SSH into a remote server e.g. an AWS EC2 server.

SSH via Jenkins

Go to “Manage Plugins” under “Manage Jenkins” on the left side of the Jenkins Dashboard

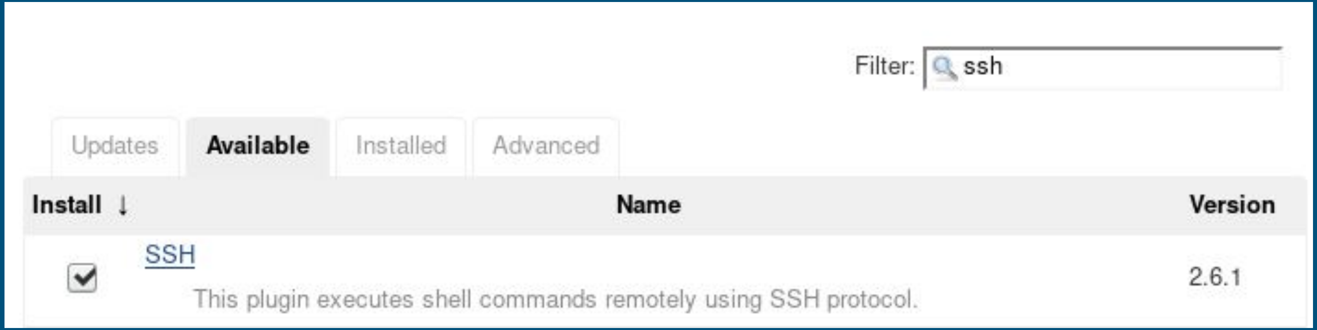


Manage Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

SSH via Jenkins

Select “Available” and search for “SSH Plugin”:

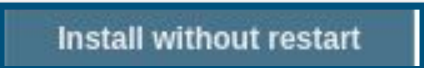


Filter: ssh

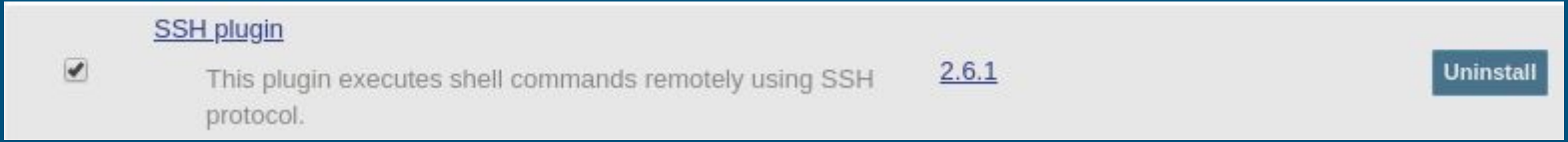
Updates | Available | Installed | Advanced

Install ↓	Name	Version
✓	SSH This plugin executes shell commands remotely using SSH protocol.	2.6.1

Select “Install without restart”.



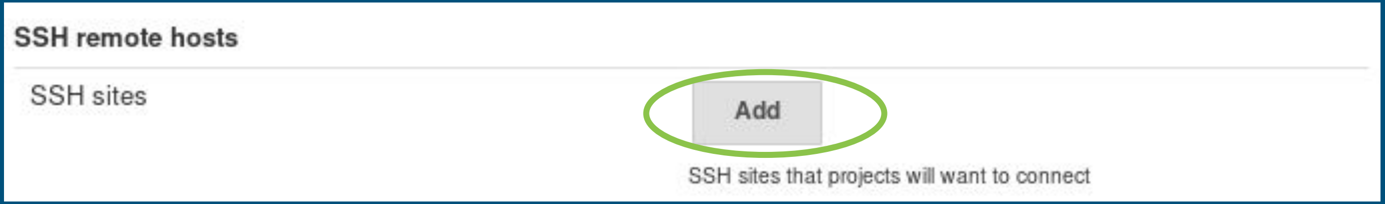
You should be able to see this plugin under the “Installed”tab:



✓	SSH plugin This plugin executes shell commands remotely using SSH protocol.	2.6.1	Uninstall

SSH via Jenkins

Select “Configure System” under “Manage Jenkins” and scroll down to find:

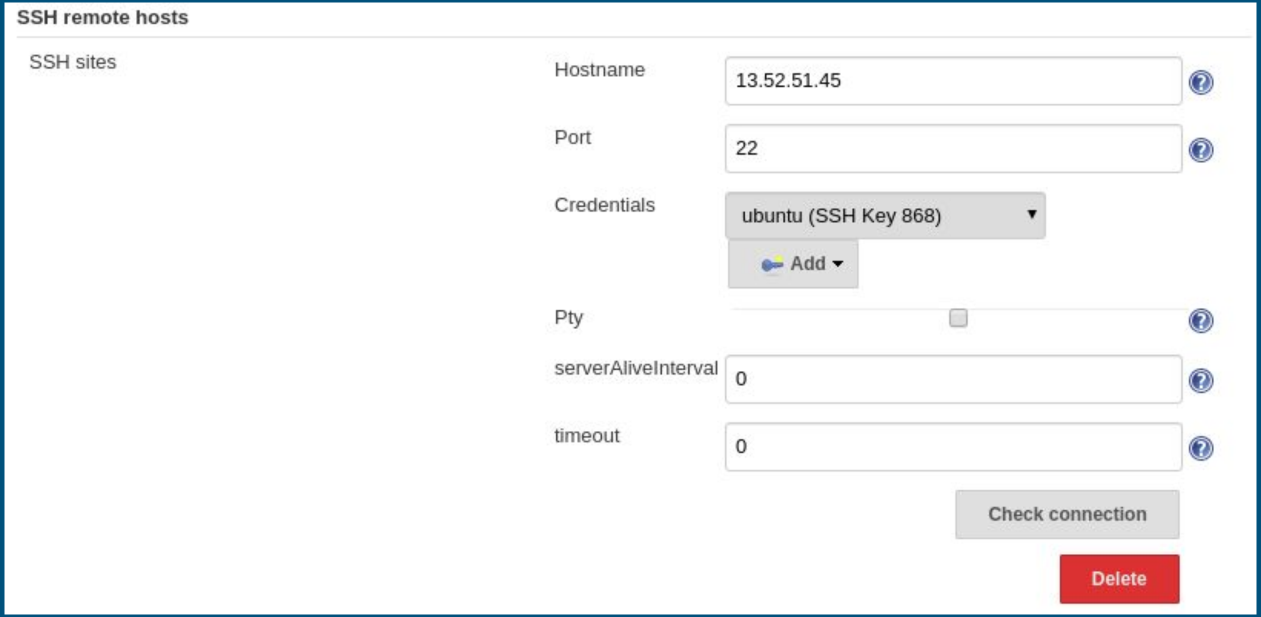

SSH remote hosts

SSH sites

SSH sites that projects will want to connect

Click on “Add”.

SSH via Jenkins



SSH remote hosts

SSH sites

Hostname	13.52.51.45
Port	22
Credentials	ubuntu (SSH Key 868)
Pty	☐
serverAliveInterval	0
timeout	0

Fill the “Hostname” with the Public IP of your EC2 instance or remote server.

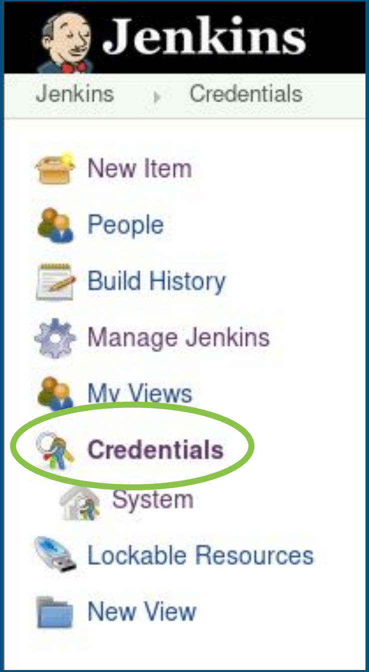
Have “Port” as 22. Port 22 is default for SSH.

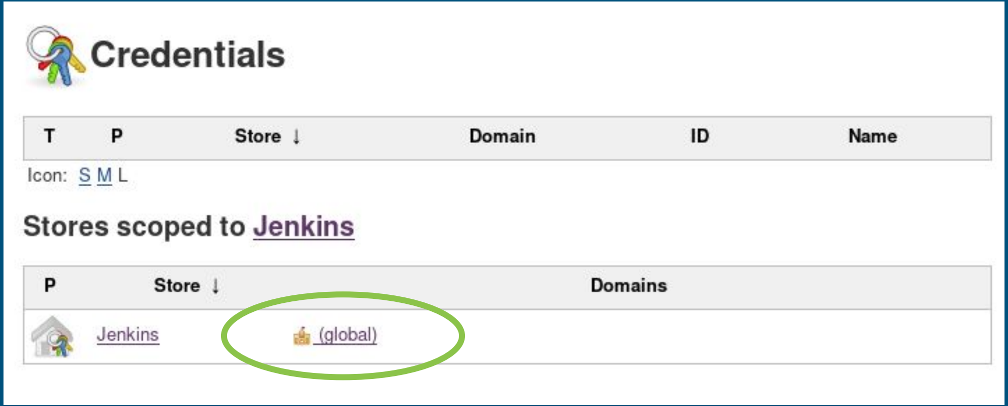
Fill in the rest of the fields as shown in the screenshot.

See the next slide for filling in the “Credentials”.

SSH via Jenkins

On the left side of the dashboard, under the same list as "Manage Jenkins", select "Credentials"





T	P	Store ↓	Domain	ID	Name

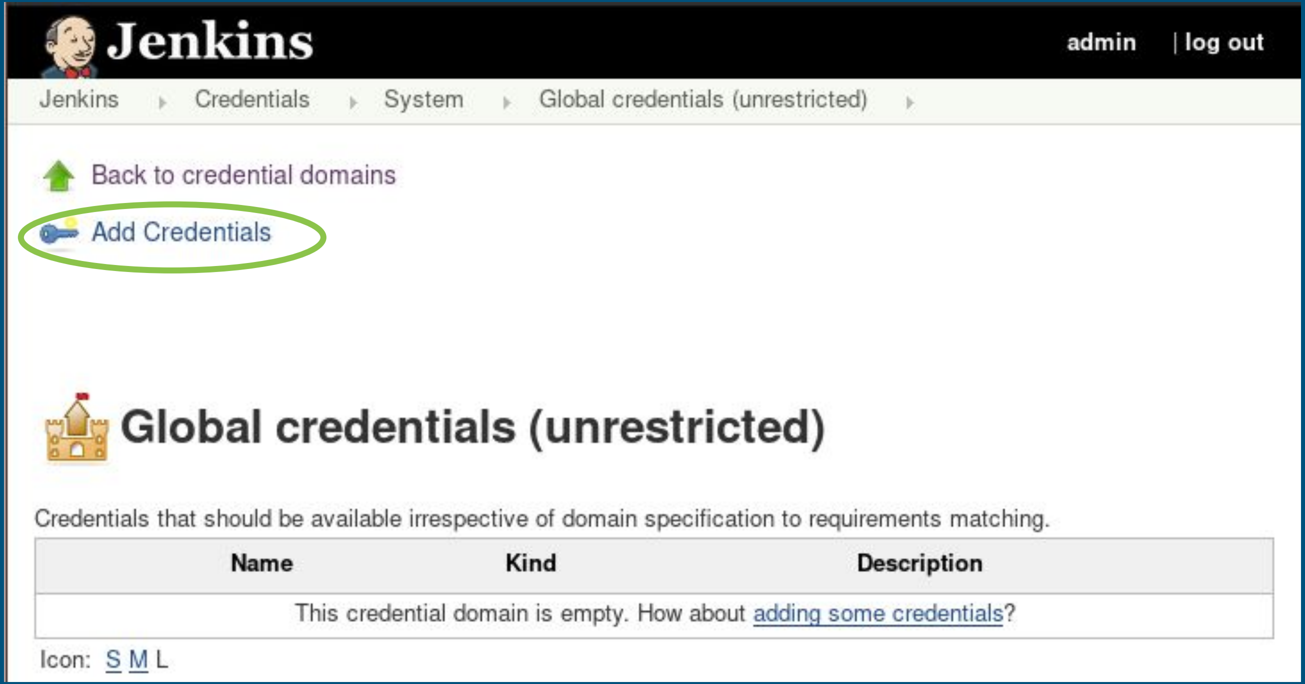
Icon: S M L

Stores scoped to Jenkins

P	Store ↓	Domains
	Jenkins	(global)

Click on "(global)" as shown in the screenshot above.

SSH via Jenkins


Jenkins admin | log out

Jenkins ▸ Credentials ▸ System ▸ Global credentials (unrestricted) ▸

Back to credential domains

Add Credentials

Global credentials (unrestricted)

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

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

Icon: S M L

Click on “Add Credentials”.

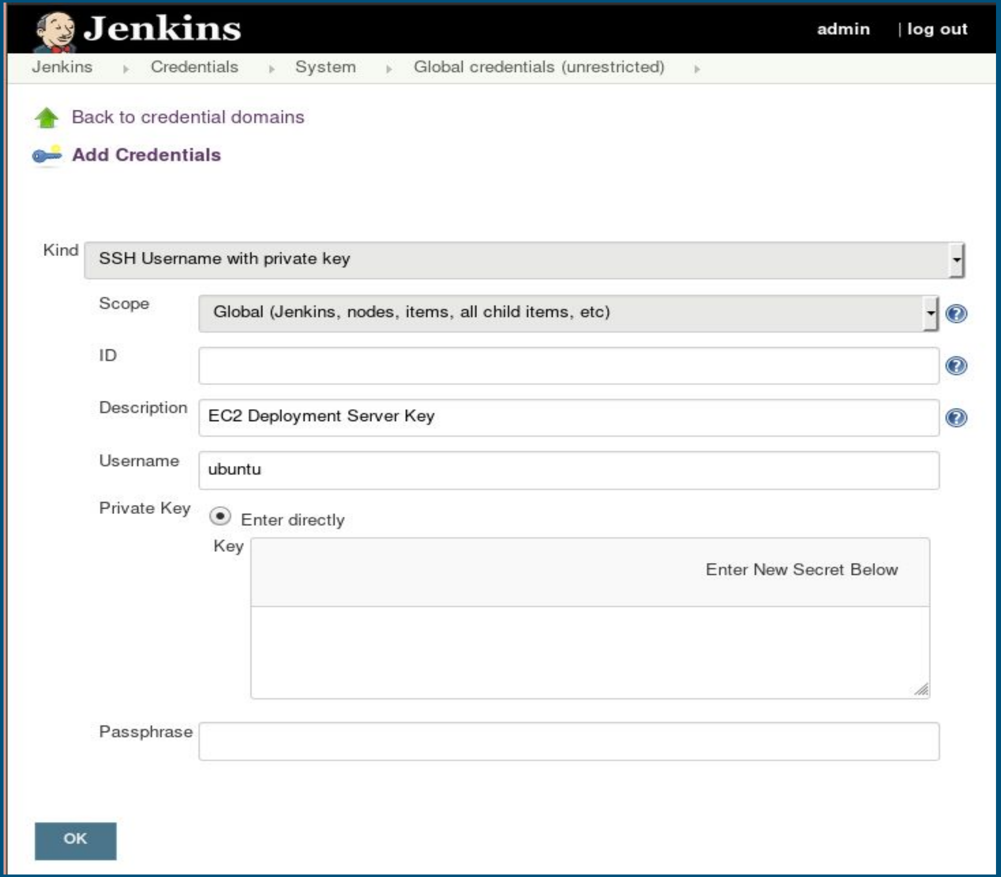
SSH via Jenkins

Select "Kind" as "SSH Username with private key".

"Username" as the username used to SSH into the remote server or EC2 instance.

The "Private Key" is the contents of the ".pem" file as provided by AWS when provisioning an EC2 instance.

Otherwise, use a private key as generated by the SSH Keygen. Do not use the same key pair as the Deploy key in slide 48.


Jenkins
admin
| log out
Jenkins › Credentials › System › Global credentials (unrestricted) ›
Back to credential domains
Add Credentials
OK

Kind	SSH Username with private key
Scope	Global (Jenkins, nodes, items, all child items, etc)
ID	
Description	EC2 Deployment Server Key
Username	ubuntu
Private Key	● Enter directly
Key	Enter New Secret Below
Passphrase	

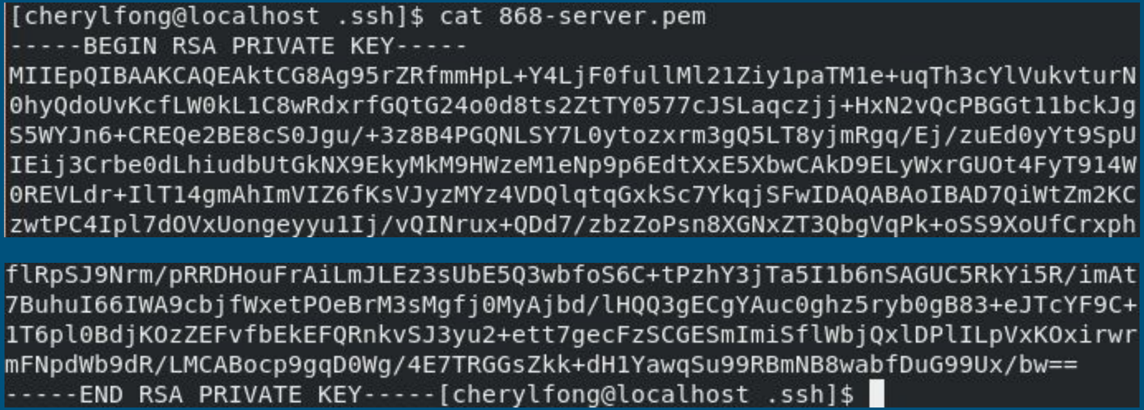
SSH via Jenkins

Select “Kind” as “SSH Username with private key”.

“Username” as the username used to SSH into the remote server or EC2 instance.

The “Private Key” is the contents of the “.pem” file as provided by AWS when provisioning an EC2 instance.

Otherwise, use a private key as generated by the SSH Keygen. Do not use the same key pair as the Deploy key in slide 48.

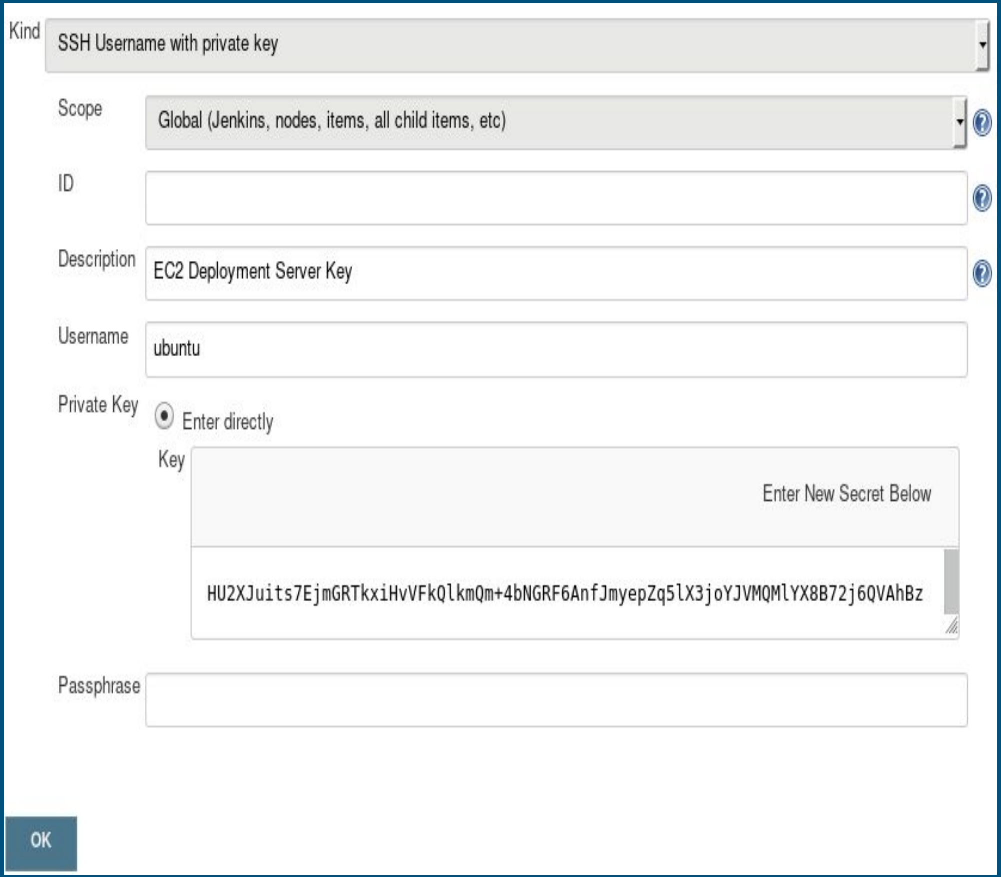


```
[cherylfong@localhost .ssh]$ cat 868-server.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAktCG8Ag95rZRfmmHpL+Y4LjF0fullMl21Ziy1paTM1e+uqTh3cYlVukvturN
0hyQdoUvKcfLW0kL1C8wRdxrfGQtG24o0d8ts2ZtTY0577cJSLaqczjj+HxN2vQcPBGGt11bckJg
S5WYJn6+CREQe2BE8cS0Jgu/+3z8B4PGQNLSY7L0ytozxrm3gQ5LT8yjmRgq/Ej/zuEd0yYt9SpU
IEij3Crbe0dLhiudbUtGkNX9EkyMkM9HWzeM1eNp9p6EdtXxE5XbwCAkD9ELyWxrGUOt4FyT914W
0REVLdr+IlT14gmAhImVIZ6fKsVJyzMYz4VDQlqtqGxkSc7YkqjSFwIDAQABAoIBAD7QiWtZm2KC
zwtPC4Ipl7dOVxUongeyyu1Ij/vQINrux+QDd7/zbzZoPsn8XGNxZT3QbgVqPk+oSS9XoUfCrxph

flRpSJ9Nrm/pRRDHouFrAiLmJLEz3sUbE5Q3wbfoS6C+tPzhY3jTa5I1b6nSAGUC5RkYi5R/imAt
7BuhuI66IWA9cbjfWxetPOeBrM3sMgfj0MyAjbd/lHQQ3gECgYAuc0ghz5ryb0gB83+eJTcYF9C+
1T6pl0BdjKOzZEFvfbEkEFQRnkvSJ3yu2+ett7gecFzSCGESmImiSflWbjQxlDPlILpVxKOxirwr
mFNpdWb9dR/LMCABocp9gqD0Wg/4E7TRGGsZkk+dH1YawqSu99RBmNB8wabfDuG99Ux/bw==
-----END RSA PRIVATE KEY-----[cherylfong@localhost .ssh]$
```

SSH via Jenkins

SSH Keys provided by AWS (i.e. .pem file) usually do not have a passphrase associated.

If your generated SSH key has a passphrase, add the passphrase to the ”Passphrase” field.



Kind: SSH Username with private key

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

ID:

Description: EC2 Deployment Server Key

Username: ubuntu

Private Key: ● Enter directly

Key: Enter New Secret Below

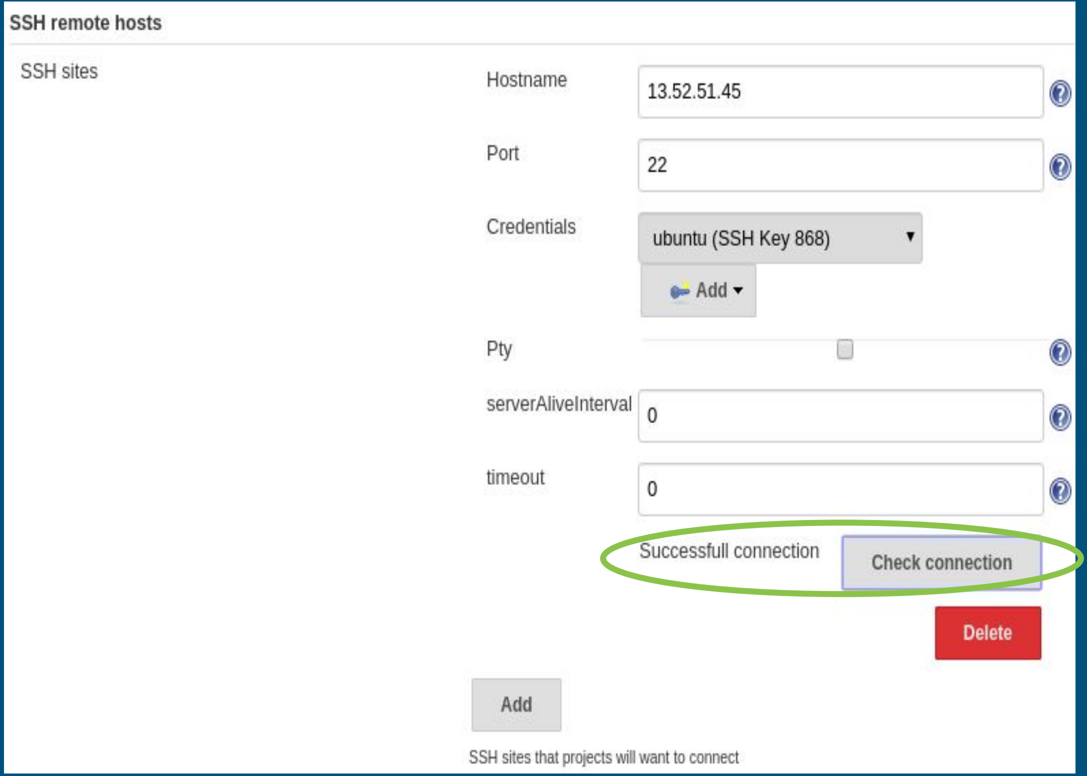
```
HU2XJuits7EjmGRTkxiHvVFkQlkmQm+4bNGRF6AnfJmyepZq5lX3joYJVMQMlYX8B72j6QVAhBz
```

Passphrase:

SSH via Jenkins

Add the “Credentials” that were just created in slide 59.

Then check the connection, it should say “Successful connection”.



SSH remote hosts

SSH sites

Hostname	13.52.51.45
Port	22
Credentials	ubuntu (SSH Key 868)
Pty	☐
serverAliveInterval	0
timeout	0

Successfull connection

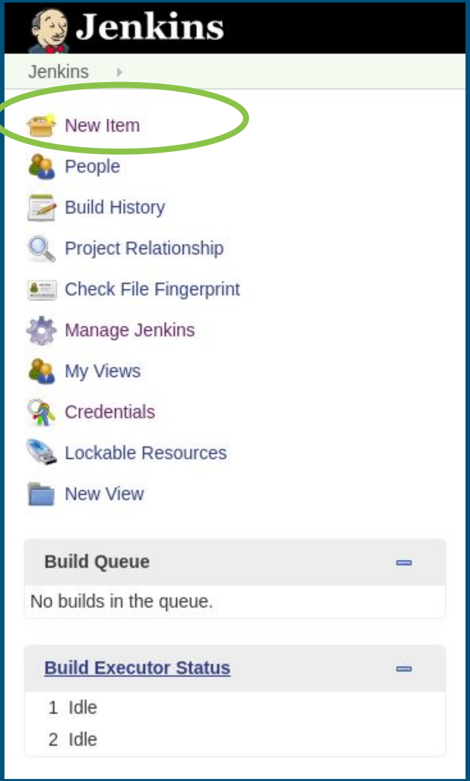
SSH sites that projects will want to connect

Deployment Example

This example, will utilize:

1. The **GitHub** section in “Configure System” (slide 32 onwards)
2. The **SSH remote hosts** section in “Configure System” (slide 50 onwards)
3. The Deploy key (slide 43 onwards)

Deployment Example



Jenkins

Jenkins

New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Credentials

Lockable Resources

New View

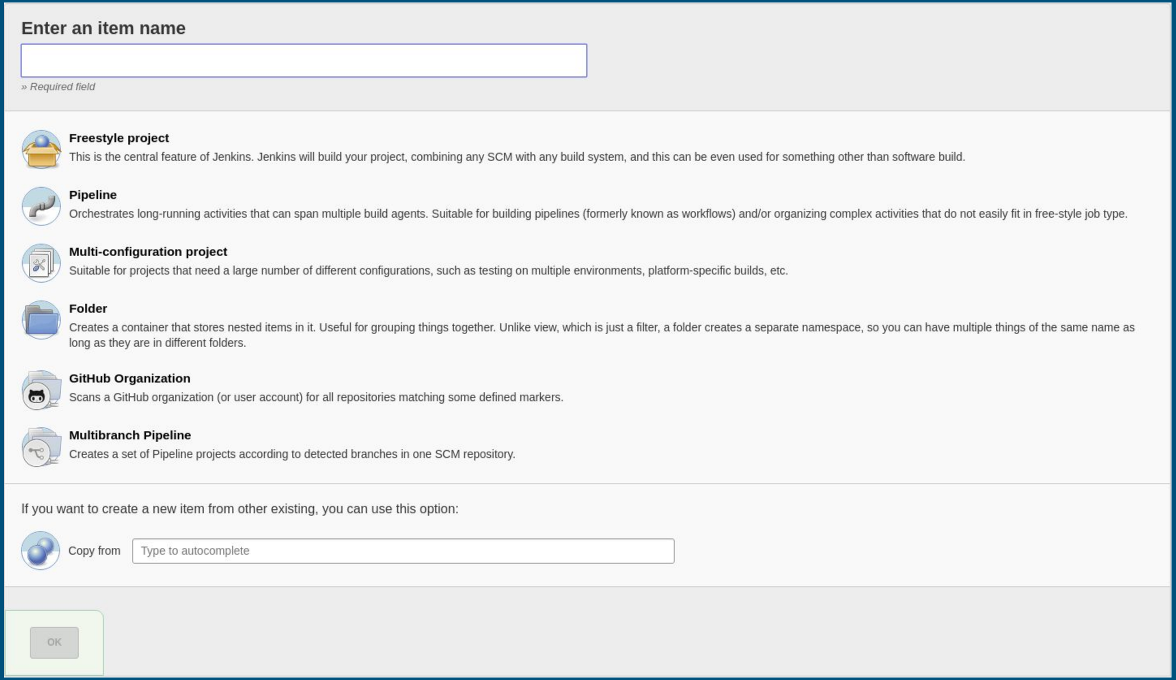
Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle



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

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

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

If you want to create a new item from other existing, you can use this option:

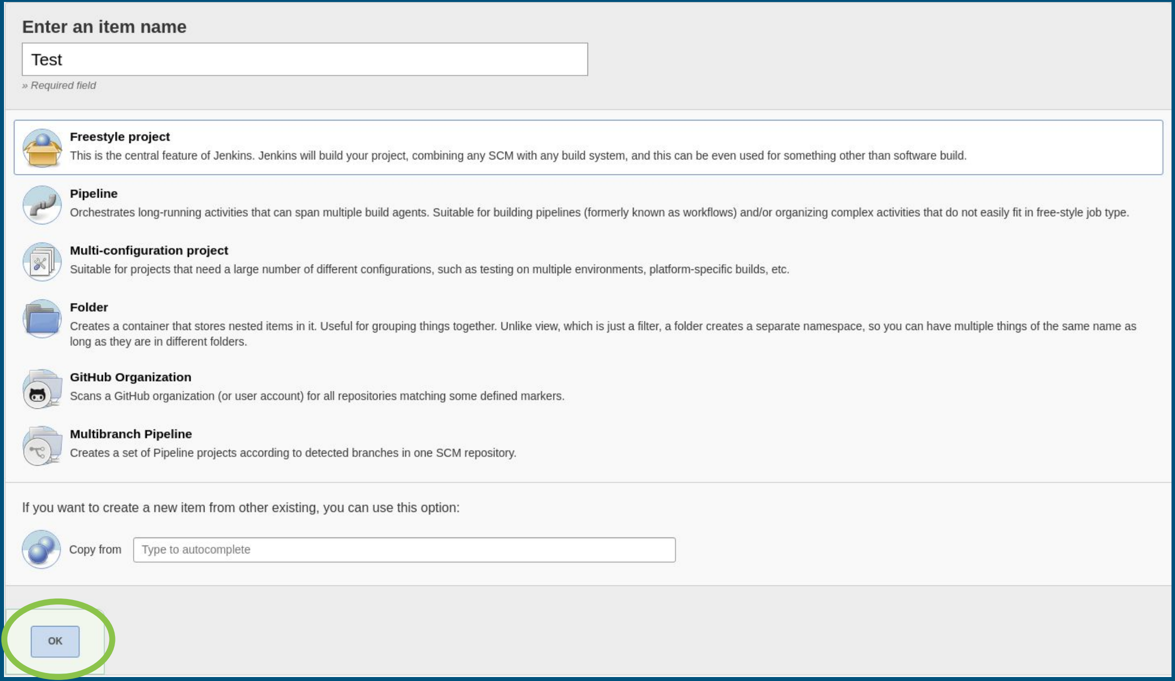
Copy from: Type to autocomplete

OK

Deployment Example

Provide a name for your deployment pipeline and select “Freestyle project”.

Click “OK” when done.


Enter an item name

Test

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

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

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

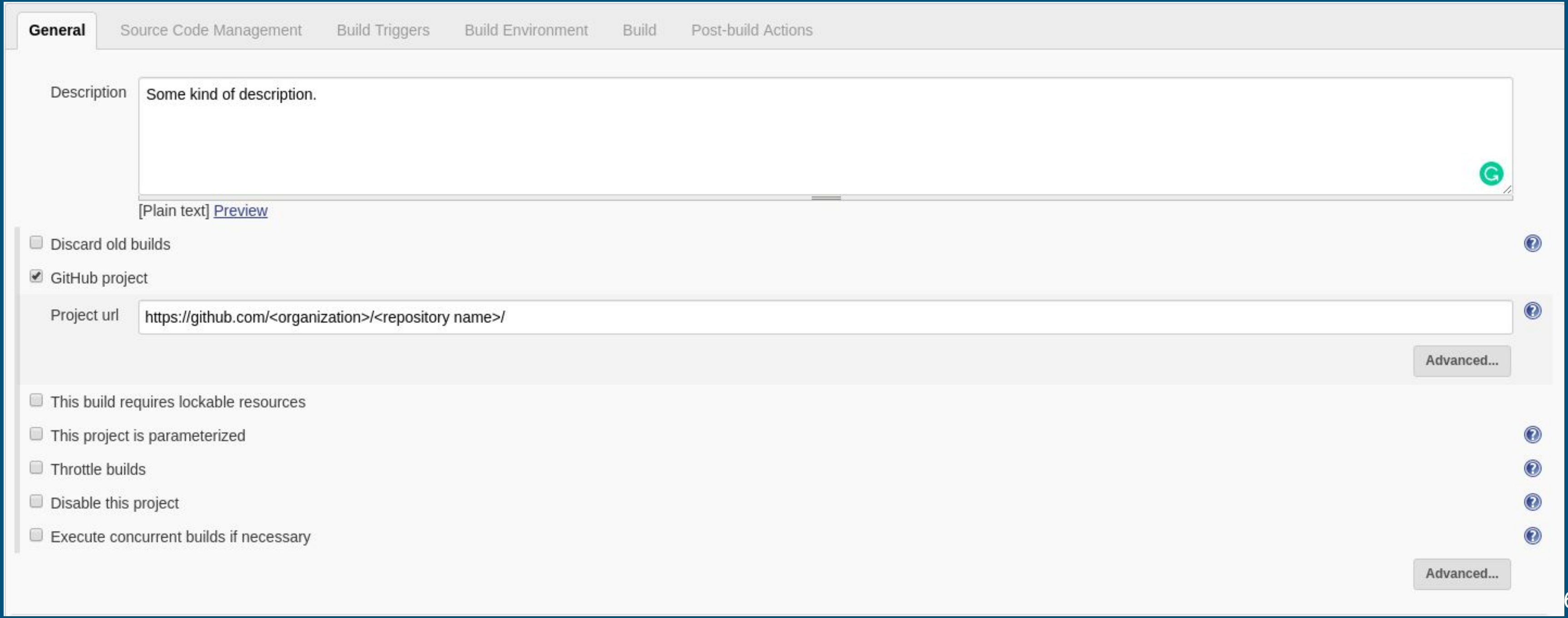
If you want to create a new item from other existing, you can use this option:

Copy from Type to autocomplete

OK

Deployment Example

Fill in the fields as shown in the screenshot below:


General | Source Code Management | Build Triggers | Build Environment | Build | Post-build Actions

Description: Some kind of description.

[Plain text] Preview

- [ ] Discard old builds
- [✓] GitHub project

Project url: https://github.com/<organization>/<repository name>/

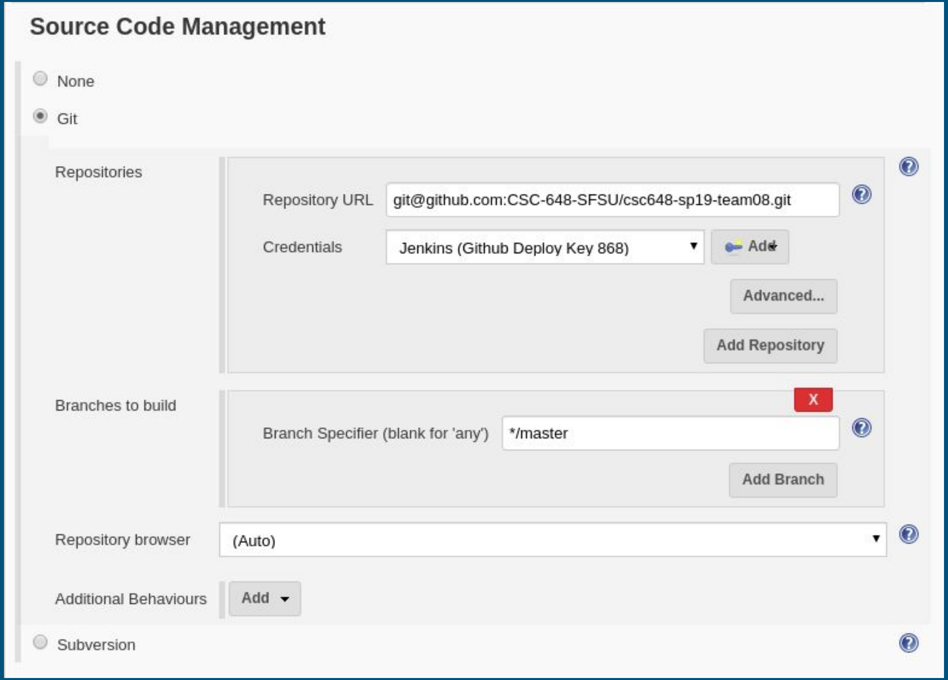
Advanced...

- [ ] This build requires lockable resources
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Disable this project
- [ ] Execute concurrent builds if necessary

Advanced...

Deployment Example

Fill in the fields as shown in the screenshot below:



Source Code Management

○ None
● Git

Repositories
- Repository URL: git@github.com:CSC-648-SFSU/csc648-sp19-team08.git
- Credentials: Jenkins (Github Deploy Key 868)

Branches to build
- Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

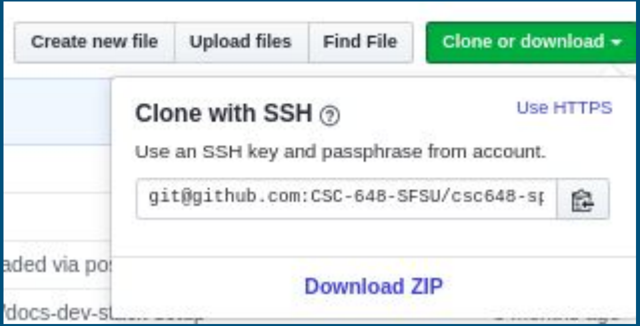
Additional Behaviours: Add

○ Subversion

Under “Credentials”, select the Deploy key as created on slide 43 onwards.

Under Repository URL:

Provide the following that can be found on the GitHub repository homepage.



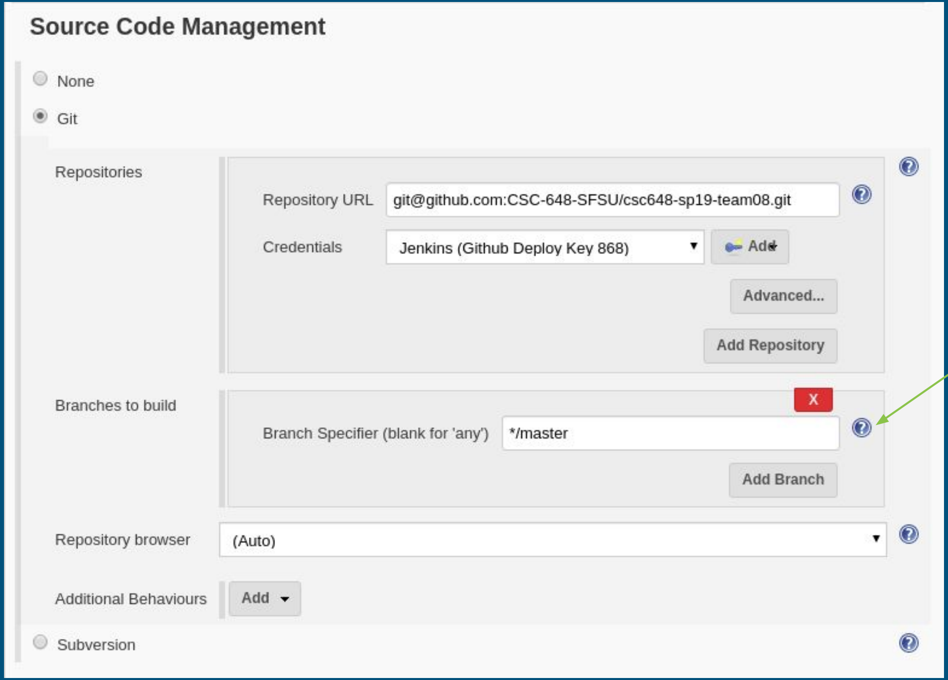
Clone with SSH

Use an SSH key and passphrase from account.

git@github.com:CSC-648-SFSU/csc648-sp

Deployment Example

Fill in the fields as shown in the screenshot below:



Source Code Management

None [✓ Git] Subversion

Repositories

Repository URL: git@github.com:CSC-648-SFSU/csc648-sp19-team08.git

Credentials: Jenkins (Github Deploy Key 868)

Branches to build

Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

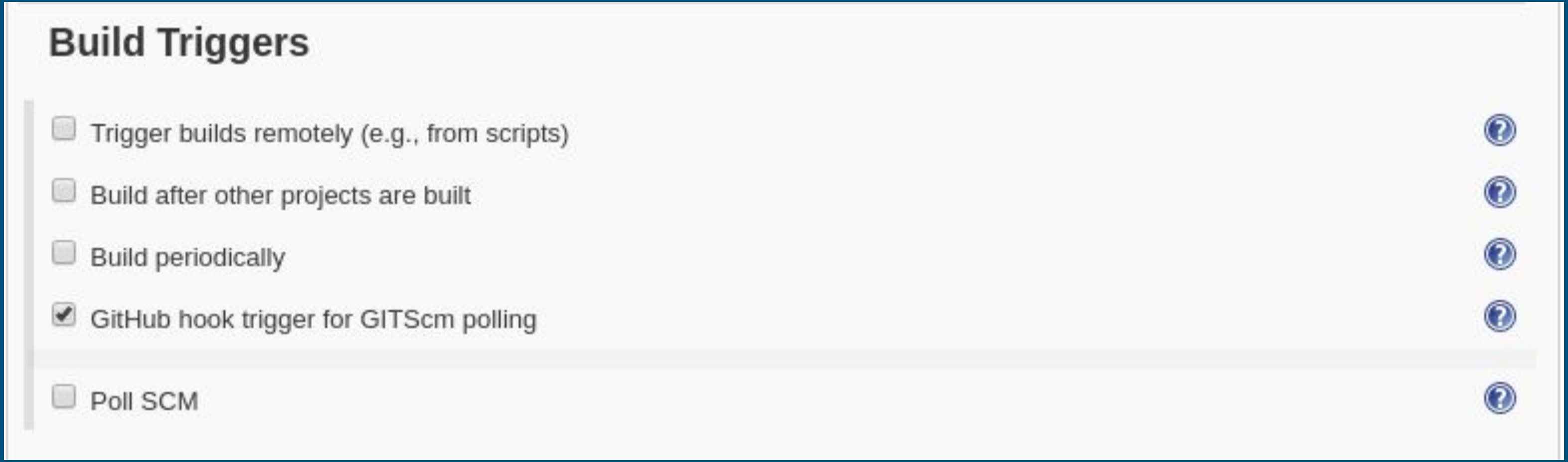
Additional Behaviours

Select the branch that you want Jenkins to build, under "Branches to build":

In the screenshot, this Jenkins Pipeline will build the "master" branch. Select the question mark for more information.

Deployment Example

Fill in the fields as shown in the screenshot below:



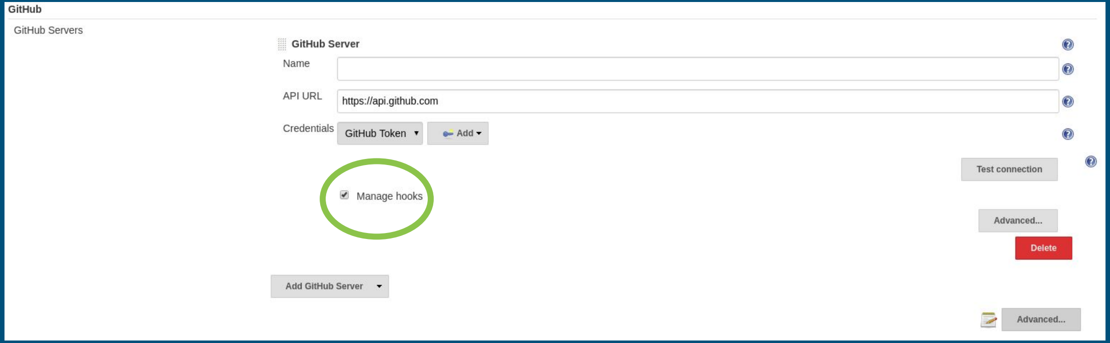
Build Triggers

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

Deployment Example

To verify that the Build Triggers works:

Manage hooks must be selected, when completing slide 4.


GitHub

GitHub Servers

GitHub Server

Name:

API URL: https://api.github.com

Credentials: GitHub Token

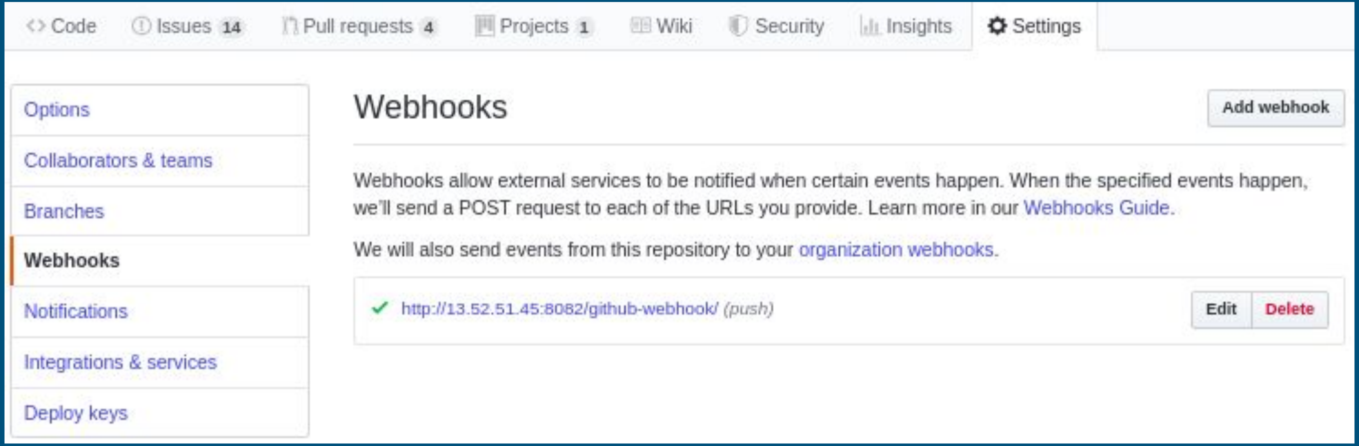
[✓ Manage hooks]

Next go to https://github.com/<organization>/<repository-name>/settings/hooks

Deployment Example

To verify that the Build Triggers works:

This Webhooks setting should be automatically populated for you when you “Save” or “Apply” the Jenkins Freestyle project.



Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

We will also send events from this repository to your organization webhooks.

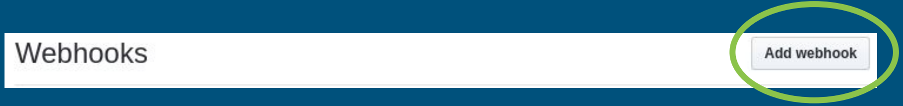
✓ http://13.52.51.45:8082/github-webhook/ *(push)*

Deployment Example

To verify that the Build Triggers works:

The Webhooks settings for your Jenkins Freestyle Project should look something like this.

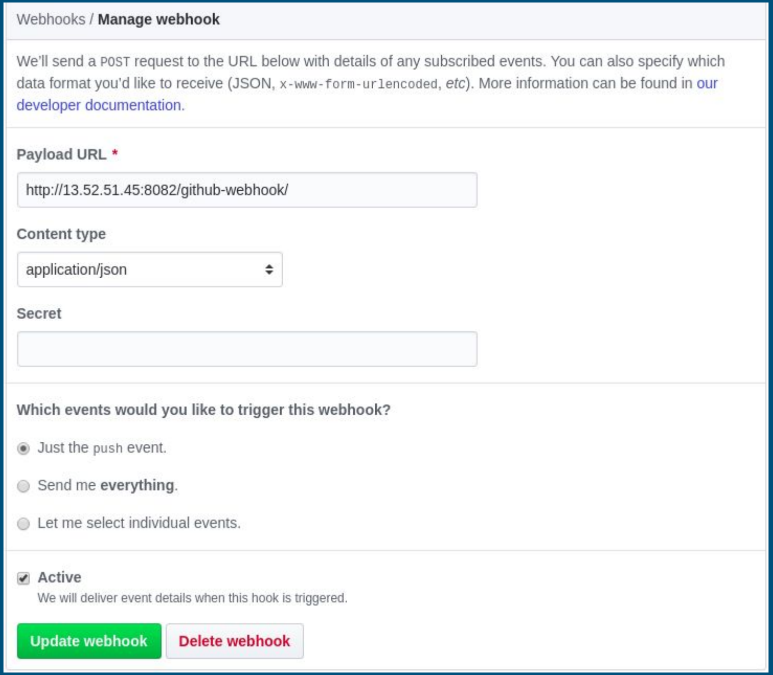
Add a new webhook if you do not see the webhook on slide 68.



Webhooks

Note the payload URL:
https://public-ip:port-number/github-webhook/

The rest of the fields in the screenshot are default settings.



Webhooks / **Manage webhook**

We'll send a `POST` request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, *etc*). More information can be found in our developer documentation.

Payload URL *
http://13.52.51.45:8082/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.

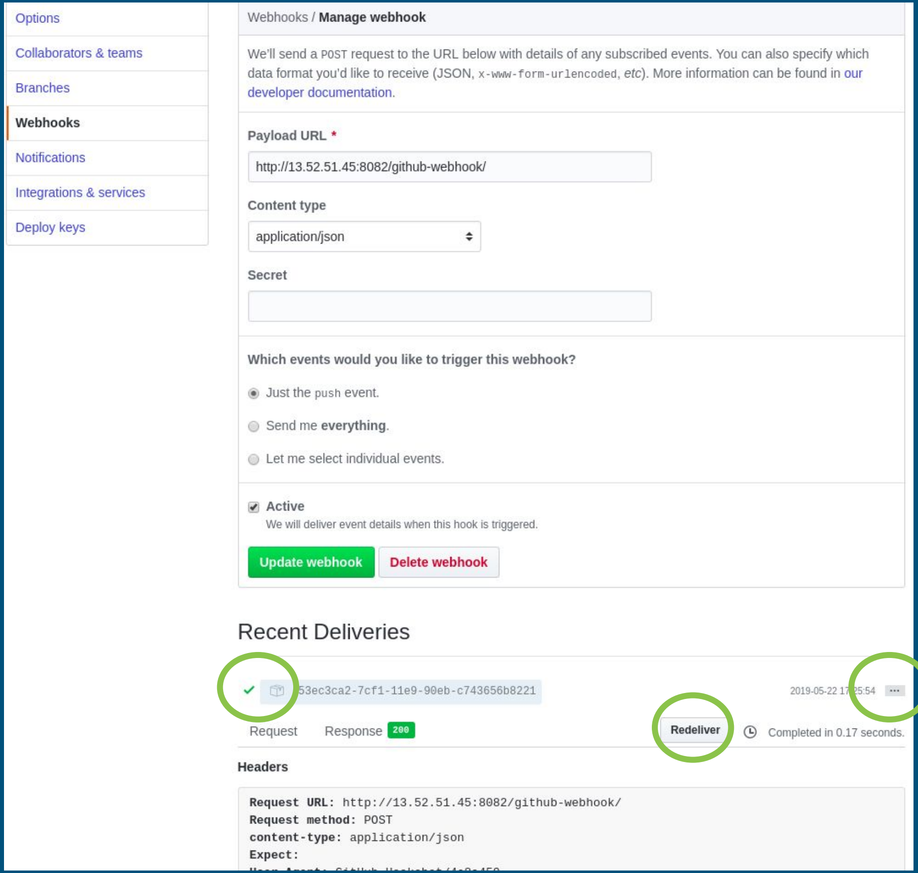
Deployment Example

To verify that the Build Triggers works:

This will confirm your settings from the previous slides are working.

Updating your webhook by clicking “Update webhook”, will deliver a payload to Jenkins. You can send another by pressing the “…” button and “Redeliver”.

A successful delivery will have a check mark shown on the right →



Webhooks / **Manage webhook**

We'll send a `POST` request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, *etc*). More information can be found in our developer documentation.

Payload URL *

http://13.52.51.45:8082/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.

Recent Deliveries

✓ 53ec3ca2-7cf1-11e9-90eb-c743656b8221 2019-05-22 17:25:54 ···

Request Response 200 Redeliver Completed in 0.17 seconds.

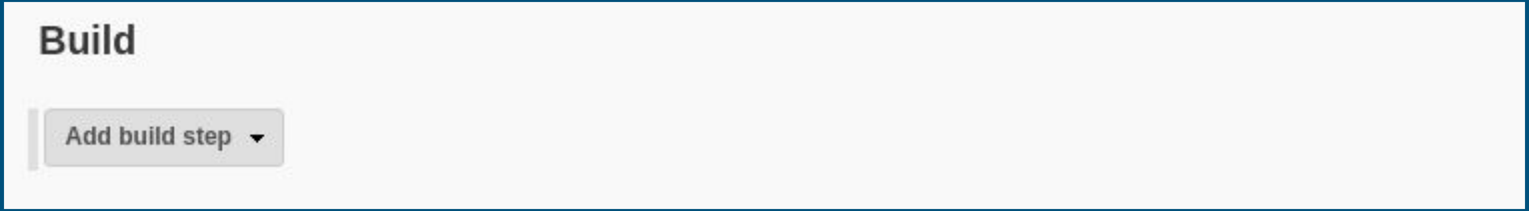
Headers

```
Request URL: http://13.52.51.45:8082/github-webhook/
Request method: POST
content-type: application/json
Expect:
```

Deployment Example

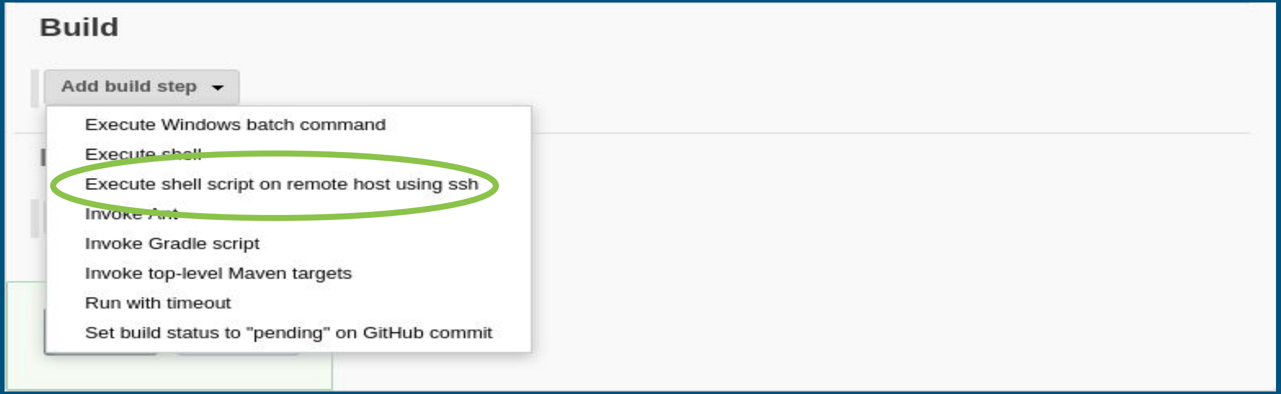
SSH into the EC2 server or remote host:

Scroll down of the Freestyle project settings to look for “Build”.



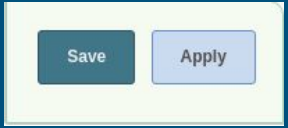
Deployment Example

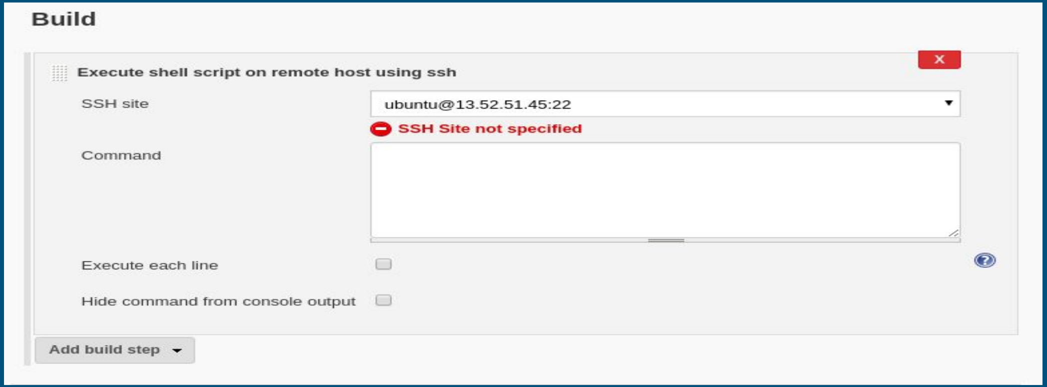
Select "Execute shell script on remote host using ssh":


Build
Add build step
Execute Windows batch command
Execute shell
Execute shell script on remote host using ssh
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit

Deployment Example

The “SSH site not specified” warning will disappear once, the “Apply” button for the Freestyle project has been clicked.




Build

Execute shell script on remote host using ssh

SSH site: ubuntu@13.52.51.45:22

SSH Site not specified

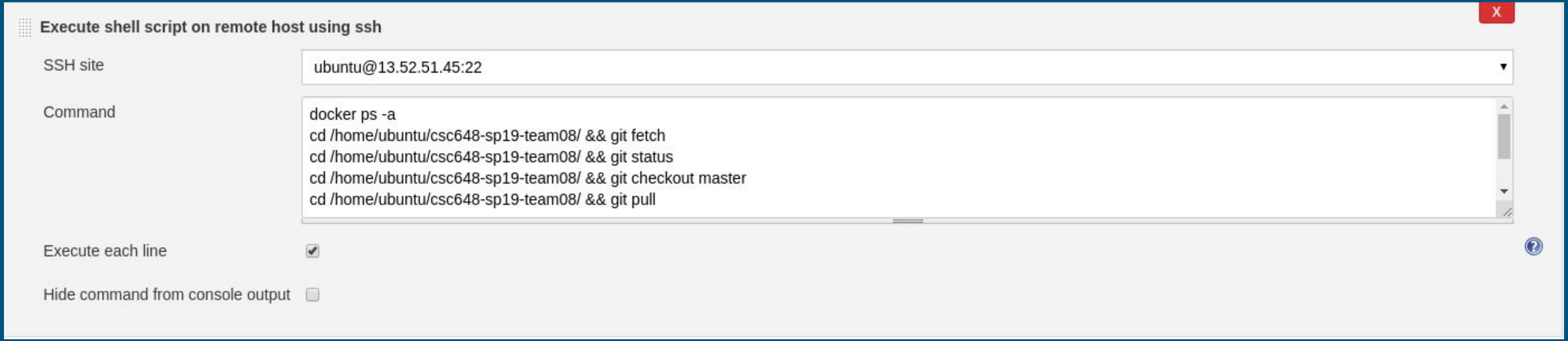
Command

Execute each line

Hide command from console output

Deployment Example

The following git commands will update git references, checkout to the master branch and then pull the latest updates on master.



Execute shell script on remote host using ssh

SSH site: ubuntu@13.52.51.45:22

Command:

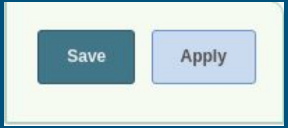
```
docker ps -a
cd /home/ubuntu/csc648-sp19-team08/ && git fetch
cd /home/ubuntu/csc648-sp19-team08/ && git status
cd /home/ubuntu/csc648-sp19-team08/ && git checkout master
cd /home/ubuntu/csc648-sp19-team08/ && git pull
```

Execute each line ✓

Hide command from console output ☐

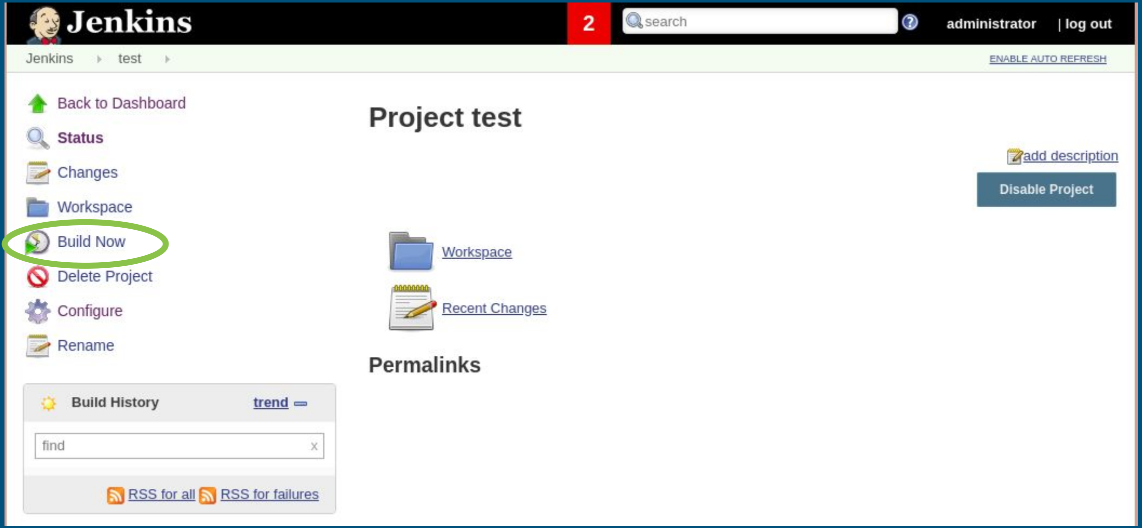
Deployment Example

Save the Freestyle project configuration by clicking “Save”.



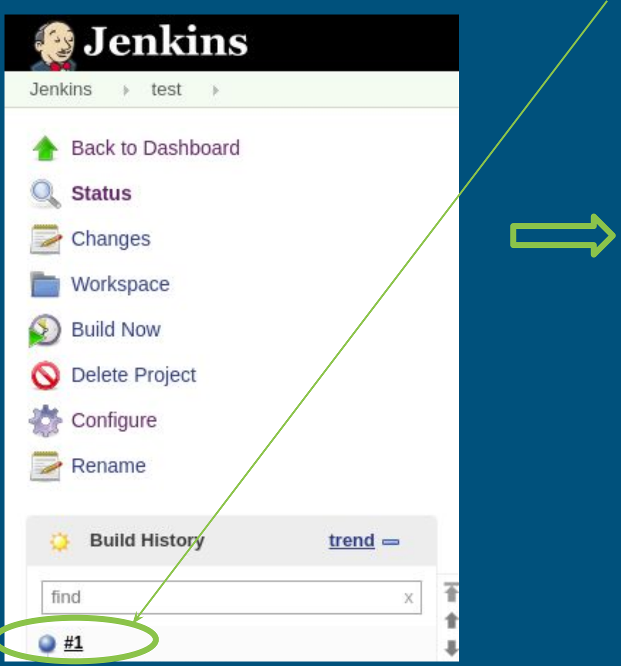
You will automatically be brought to the Project dashboard.

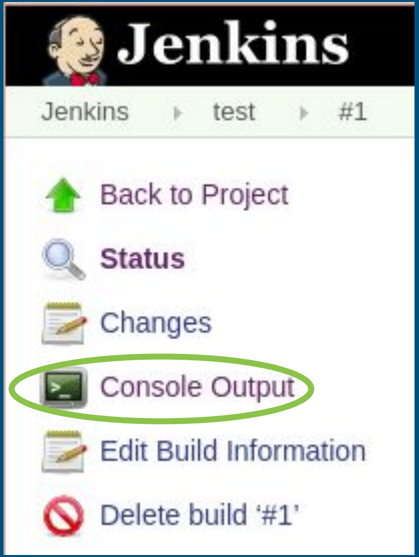
Click “Build Now”.



Deployment Example

After clicking “Build Now”, you’ll see build #1:

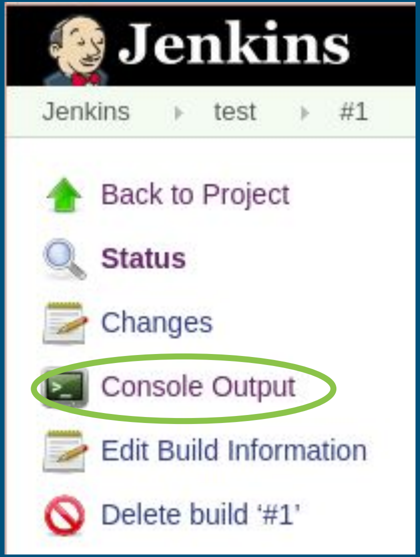


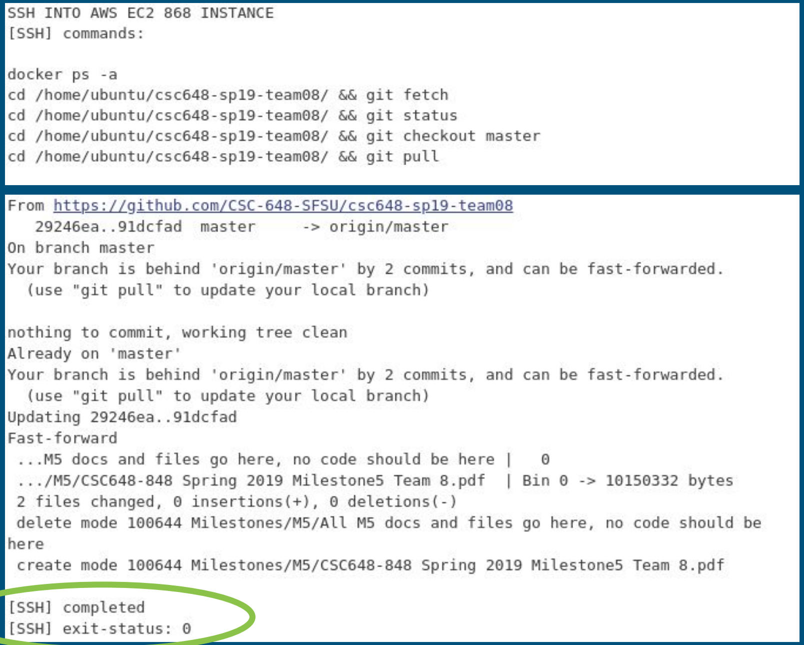


Click on the “Console Output”.

Deployment Example

The “Console Output” should look something like this:



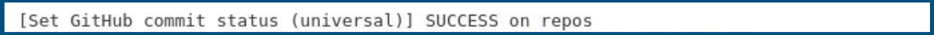


```
SSH INTO AWS EC2 868 INSTANCE
[SSH] commands:

docker ps -a
cd /home/ubuntu/csc648-sp19-team08/ && git fetch
cd /home/ubuntu/csc648-sp19-team08/ && git status
cd /home/ubuntu/csc648-sp19-team08/ && git checkout master
cd /home/ubuntu/csc648-sp19-team08/ && git pull
```

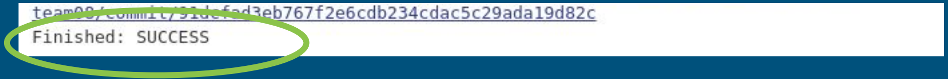
```
From https://github.com/CSC-648-SFSU/csc648-sp19-team08
   29246ea..91dcfad  master     -> origin/master
On branch master
Your branch is behind 'origin/master' by 2 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean
Already on 'master'
Your branch is behind 'origin/master' by 2 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)
Updating 29246ea..91dcfad
Fast-forward
 ...M5 docs and files go here, no code should be here |   0
 .../M5/CSC648-848 Spring 2019 Milestone5 Team 8.pdf  | Bin 0 -> 10150332 bytes
 2 files changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 Milestones/M5/All M5 docs and files go here, no code should be
here
 create mode 100644 Milestones/M5/CSC648-848 Spring 2019 Milestone5 Team 8.pdf

[SSH] completed
[SSH] exit-status: 0
```

Deployment Example

The “Console Output” log continued:

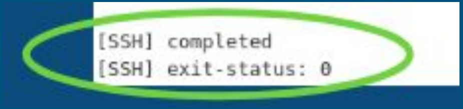


```
[Set GitHub commit status (universal)] SUCCESS on repos
```



```
team08/commit/91defad3eb767f2e6cdb234cdac5c29ada19d82c
Finished: SUCCESS
```

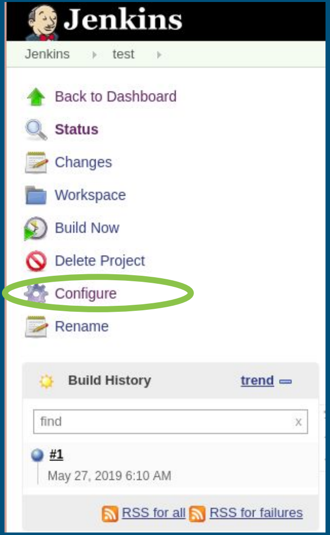
The “Finished: SUCCESS” indicates that this build is successful. All SSH Commands must return 0, for an entire build to be successful. As shown on slide 78:



```
[SSH] completed
[SSH] exit-status: 0
```

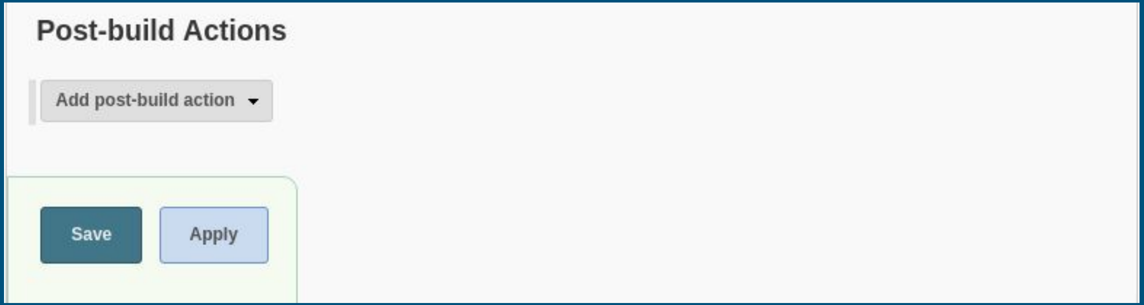
Deployment Example

Posting Build Statuses on Github:



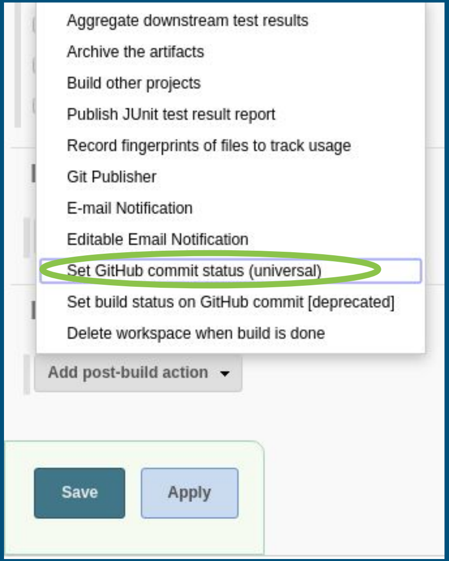
Go back to the Project's dashboard and select "Configure".

Scroll down to find "Post-build Actions":



Deployment Example

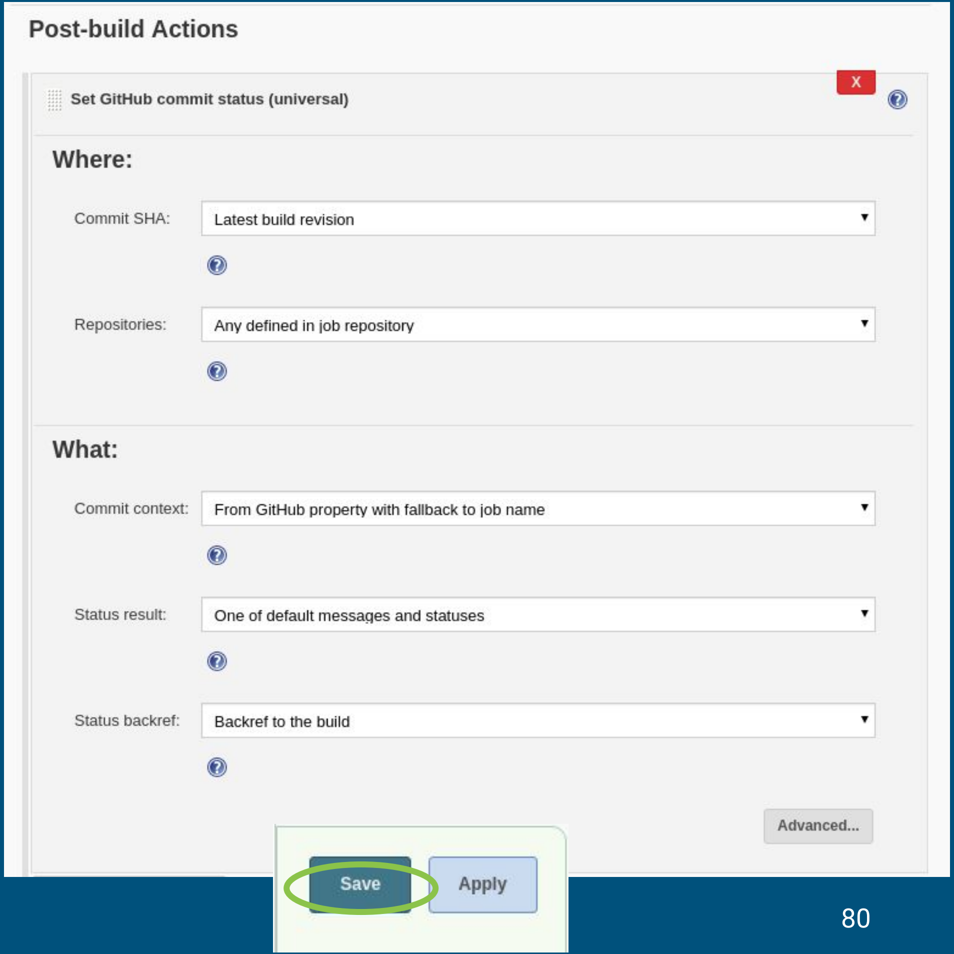
Posting Build Statuses on Github:


Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Record fingerprints of files to track usage
Git Publisher
E-mail Notification
Editable Email Notification
Set GitHub commit status (universal)
Set build status on GitHub commit [deprecated]
Delete workspace when build is done
Add post-build action
Save
Apply

Select "Set Github commit status (universal)".

Fill in the fields as shown in the screenshot. These are default settings.

Click "Save" when complete.



Where:

Commit SHA: Latest build revision

Repositories: Any defined in job repository

What:

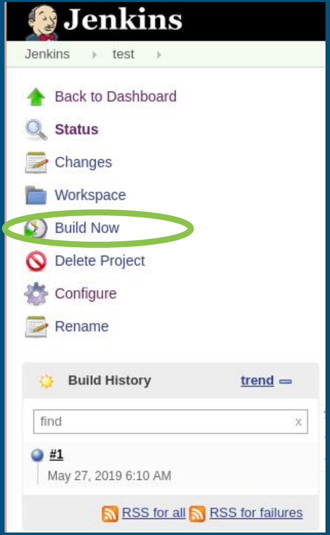
Commit context: From GitHub property with fallback to job name

Status result: One of default messages and statuses

Status backref: Backref to the build

Deployment Example

Posting Build Status on Github:



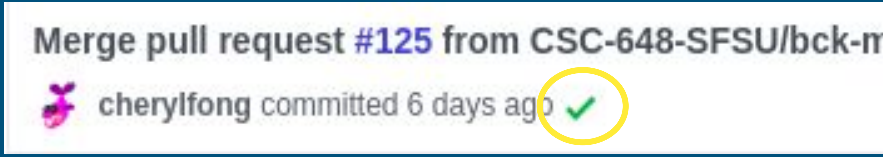
Click "Build Now" to initiate a build.

Upon completion of the build you should see a checkmark or a crossmark next to the latest commit on your GitHub repository.

Go to the repository's commit history at:

https://github.com/<organization>/<repositoty-name>/commits/master

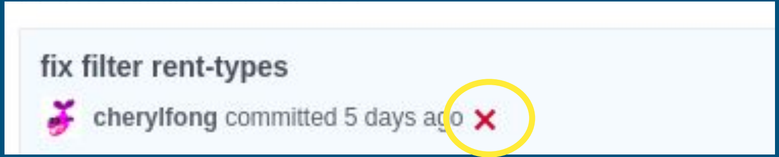
Successful Build:



Merge pull request #125 from CSC-648-SFSU/bck-m

cherylfong committed 6 days ago ✓

Unsuccessful Build:



fix filter rent-types

cherylfong committed 5 days ago ✗

FIN

Cheryl Fong

- View this slidedeck on Google slides
- For questions or suggestions please email: cheryl.fong@qq.com