**What is Jenkin for Continuous Integration?**

**What is Continuous Integration?**

In Continuous Integration after a code commit, the software is built and tested immediately. In a large project with many developers, commits are made many times during a day. With each commit code is built and tested. If the test is passed, build is tested for deployment. If deployment is a success, the code is pushed to production. This commit, build, test, and deploy is a continuous process and hence the name continuous integration/deployment.

A Continuous Integration Pipeline is a powerful instrument that consists of a set of tools designed to **host**, **monitor**, **compile** and **test** code, or code changes, like:

* **Continuous Integration Server** (Jenkins, Bamboo, CruiseControl, TeamCity, and others)
* **Source Control Tool** (e.g., CVS, SVN, GIT, Mercurial, Perforce, ClearCase and others)
* **Build tool** (Make, ANT, Maven, Ivy, Gradle, and others)
* **Automation testing framework** (Selenium, Appium, TestComplete, UFT, and others)

In this tutorial, you will learn

* [What is Continuous Integration?](https://www.guru99.com/jenkin-continuous-integration.html#1)
* [What is Jenkin?](https://www.guru99.com/jenkin-continuous-integration.html#2)
* [Jenkin History](https://www.guru99.com/jenkin-continuous-integration.html#3)
* [Why Continuous Integration with Jenkins?](https://www.guru99.com/jenkin-continuous-integration.html#4)
* [Real-world case study of Continuous Integration](https://www.guru99.com/jenkin-continuous-integration.html#5)
* [Advantages of using Jenkins](https://www.guru99.com/jenkin-continuous-integration.html#6)
* [Disadvantages of using Jenkins](https://www.guru99.com/jenkin-continuous-integration.html#7)

**What is Jenkin?**

**Jenkins** is an open source Continuous Integration server capable of orchestrating a chain of actions that help to achieve the Continuous Integration process (and not only) in an automated fashion.

Jenkins is entirely written in Java. Jenkins is a widely used application around the world that has around 300k installations and growing day by day.

It is a server-based application and requires a web server like Apache Tomcat. The reason Jenkins became so popular is that of its monitoring of repeated tasks which arise during the development of a project. For example, if your team is developing a project, Jenkins will continuously test your project builds and show you the errors in early stages of your development.

By using Jenkins, software companies can accelerate their software development process, as Jenkins can automate build and test at a rapid rate. Jenkins supports the complete development lifecycle of software from building, testing, documenting the software, deploying and other stages of a software development lifecycle.

**Jenkin History**

* Kohsuke Kawaguchi, a Java developer, working at SUN Microsystems, was tired of building the code and fixing errors repetitively. In 2004, created an automation server called Hudson that automates build and test task.
* In 2011, Oracle who owned Sun Microsystems had a dispute with Hudson open source community, so the forked Hudson and renamed it as Jenkins.
* Both Hudson and Jenkins continued to operate independently. But in short span of time, Jenkins acquired a lot of projects and contributors while Hudson remained with only 32 projects. With time, Jenkins became more popular, and Hudson is not maintained anymore.

**Why Continuous Integration with Jenkins?**

Some people might think that the old-fashioned way of developing the software is the better way. Let's understand the advantages of CI with Jenkins with the following example

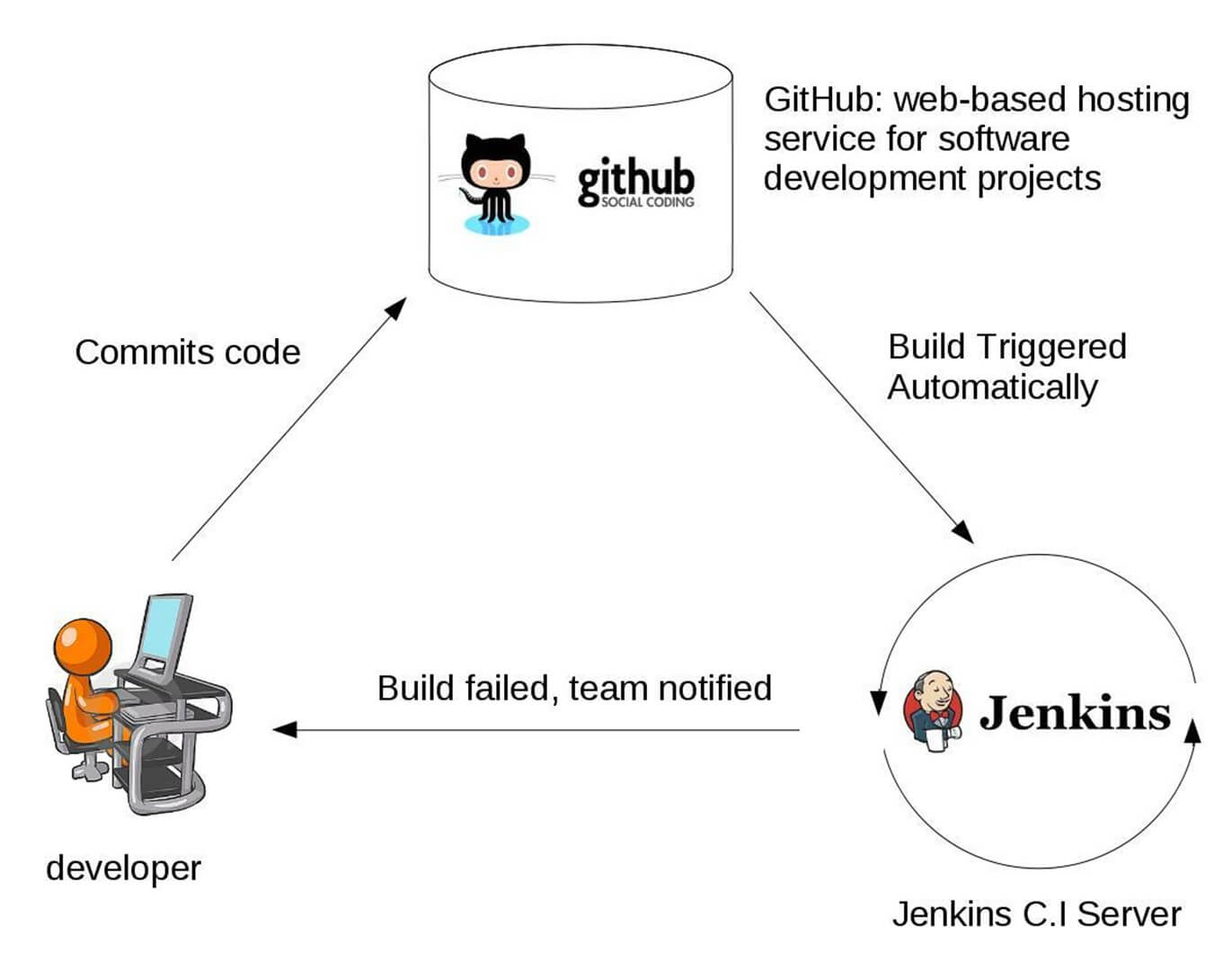
Let us imagine, that there are around 10 developers who are working on a shared repository. Some developer completes their task in 25 days while others take 30 days to complete.

|  |  |
| --- | --- |
| **Before Jenkins** | **After Jenkins** |
| Once all Developers had completed their assigned coding tasks, they used to commit their code all at same time. Later, Build is tested and deployed.  Code commit built, and test cycle was very infrequent, and a single build was done after many days. | The code is built and test as soon as Developer commits code. Jenkin will build and test code many times during the day  If the build is successful, then Jenkins will deploy the source into the test server and notifies the deployment team.  If the build fails, then Jenkins will notify the errors to the developer team. |
| Since the code was built all at once, some developers would need to wait until other developers finish coding to check their build | The code is built immediately after any of the Developer commits. |
| It is not an easy task to isolate, detect, and fix errors for multiple commits. | Since the code is built after each commit of a single developer, it's easy to detect whose code caused the built to fail |
| Code build and test process are entirely manual, so there are a lot of chances for failure. | Automated build and test process saving timing and reducing defects. |
| The code is deployed once all the errors are fixed and tested. | The code is deployed after every successful build and test. |
| Development Cycle is slow | The development cycle is fast. New features are more readily available to users. Increases profits. |

## Real-world case study of Continuous Integration

I am sure all of you aware of old phone Nokia. Nokia used to implement a procedure called nightly build. After multiple commits from diverse developers during the day, the software built every night. Since the software was built only once in a day, it's a huge pain to isolate, identify, and fix the errors in a large code base.

Later, they adopted Continuous Integration approach. The software was built and tested as soon as a developer committed code. If any error is detected, the respective developer can quickly fix the defect.

[](https://www.guru99.com/images/1/063018_1012_WhatisJenki1.jpg)

**Jenkins Plugins**

By default, Jenkins comes with a limited set of features. If you want to integrate your Jenkins installation with version control tools like Git, then you need to install plugins related to Git. In fact, for integration with tools like Maven, Amazon EC2, you need to install respective plugins in your Jenkins.

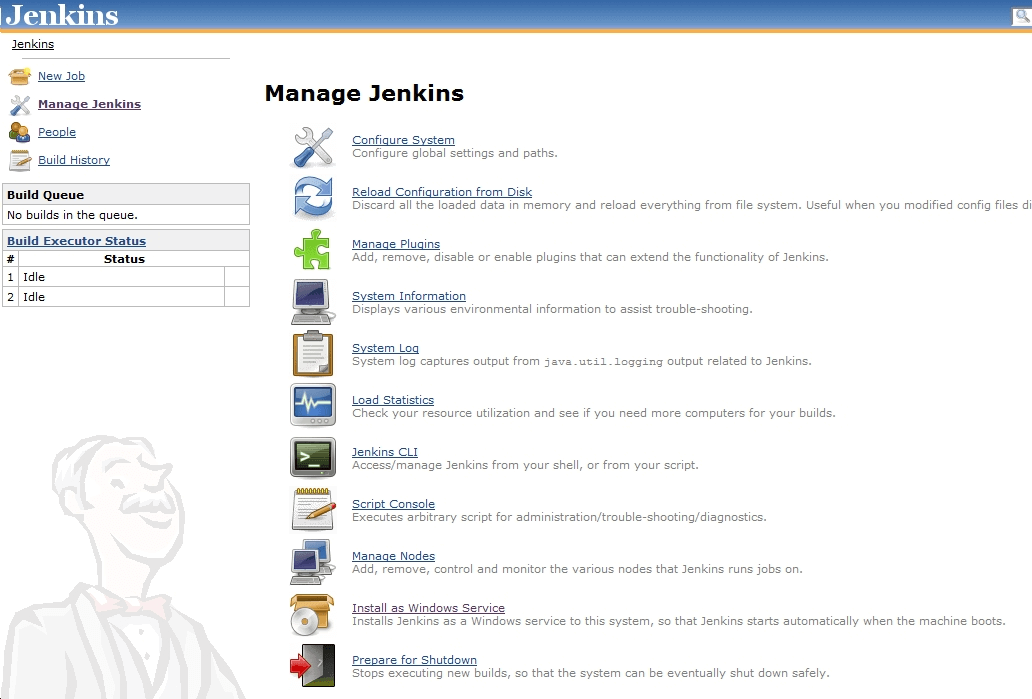
[](https://www.guru99.com/images/1/063018_1012_WhatisJenki2.png)

Figure 1Plugins integration in Jenkins

**Advantages of using Jenkins**

* Jenkins is being managed by the community which is very open. Every month, they hold public meetings and take inputs from the public for the development of Jenkins project.
* So far around 280 tickets are closed, and the project publishes stable release every three months.
* As technology grows, so does Jenkins. So far Jenkins has around 320 plugins published in its plugins database. With plugins, Jenkins becomes even more powerful and feature rich.
* Jenkins also supports cloud-based architecture so that you can deploy Jenkins in cloud-based platforms.
* The reason why Jenkins became popular is that it was created by a developer for developers.

**Disadvantages of using Jenkins**

Though Jenkins is a very powerful tool, it has its flaws.

* Its interface is out dated and not user friendly compared to current UI trends.
* Though Jenkins is loved by many developers, it's not that easy to maintain it because Jenkins runs on a server and requires some skills as server administrator to monitor its activity.
* One of the reasons why many people don't implement Jenkins is due to its difficulty in installing and configuring Jenkins.
* Continuous integrations regularly break due to some small setting changes. Continuous integration will be paused and therefore requires some developer attention.

**Conclusion:**

* In Continuous Integration, after a code commit, the software is built and tested immediately
* Jenkins is an open source Continuous Integration server capable of orchestrating a chain of actions
* Before Jenkins when all Developers had completed their assigned coding tasks, they used to commit their code all at same time. Later, Build is tested and deployed.
* After Jenkins the code is built and test as soon as Developer commits code. Jenkin will build and test code many times during the day
* By default, Jenkins comes with a limited set of features. If you want to integrate your Jenkins installation with version control tools like Git, then you need to install plugins related to Git
* The biggest pros of Jenkins is that it is managed by the community which holds public meetings and take inputs from the public for the development of Jenkins projects
* The biggest con of Jenkin is that Its interface is out dated and not user friendly compared to current UI trends.

# How to Download & Install Jenkins on Windows

Jenkins may be installed on either Windows or Unix platforms, but we will focus on Windows installation only.

**Prerequisites:**

Before you proceed to install Jenkins in your windows system, there are some prerequisites for Jenkins to install Jenkins in your computer.

**Hardware requirements:**

* You need minimum 256 MB of RAM in your computer or laptop to install Jenkins
* You need at least 1 GB of space in your hard drive for Jenkins.

**Software Requirements:**

* Since Jenkins runs on Java, you need either Java Development Kit (JDK) or Java Runtime Environment (JRE).

**Release Types**

Jenkins releases two types of versions based on the organization needs.

* Long-term support release
* Weekly release

**Long term support release (LTS) :**

Long-term support releases are available every 12 weeks. They are stable and are widely tested. This release is intended for end users.

**Weekly release:**

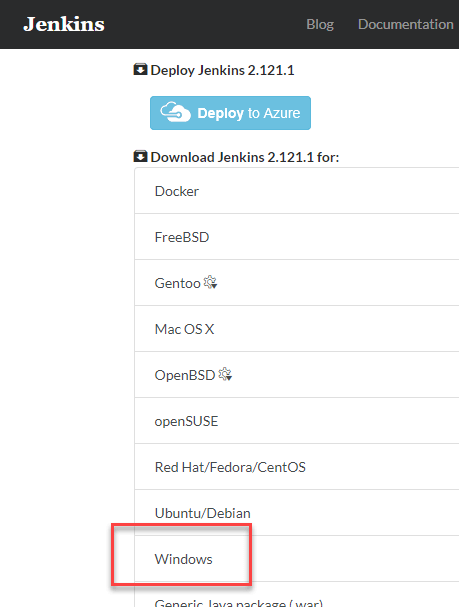
Weekly releases are made available every week by fixing bugs in its earlier version. These releases are intended towards plugin developers.

We will use the LTS release though the process remains the same for Weekly release.

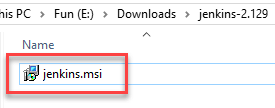
**How to Download Jenkins?**

Following steps should be followed so that to install Jenkins successfully:

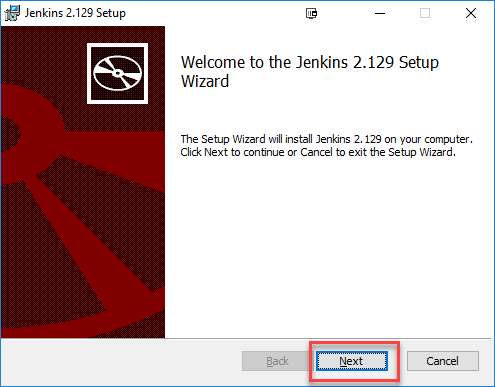
**Step 1)** Got to <https://jenkins.io/download/> and select the platform. In our case Windows

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo1.png)

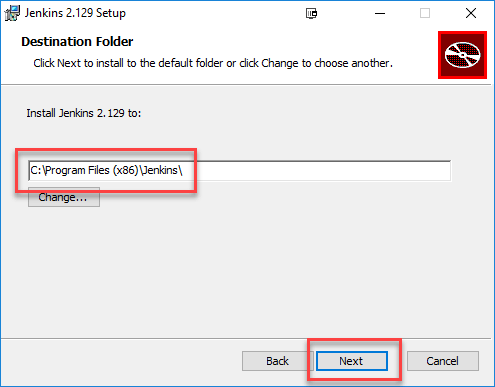
**Step 2)** Go to download location from local computer and unzip the downloaded package. Double-click on unzipped **jenkins.msi**

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo2.png)

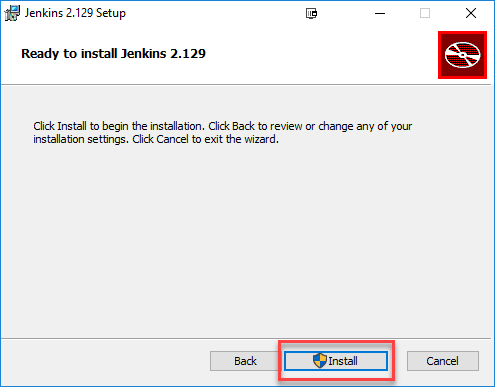
**Step 3)** In the setup screen, click Next.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo3.png)

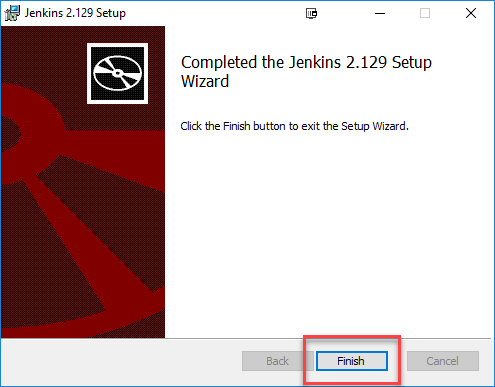
**Step 4)** Choose the location where you want to have the Jenkins instance installed (default location is C:\Program Files (x86)\Jenkins), then click on **Next** button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo4.png)

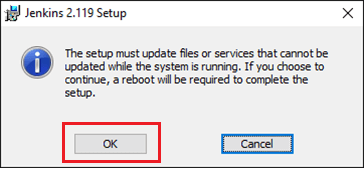
**Step 5)**Click on the Install button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo5.png)

**Step 6)** Once install is complete, click Finish.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo6.png)

**Step 7)** During the installation process an info panel may pop-up to inform the user that for a complete setup, the system should be rebooted at the end of the current installation. Click on OK button when the Info panel is popping-up:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo7.png)

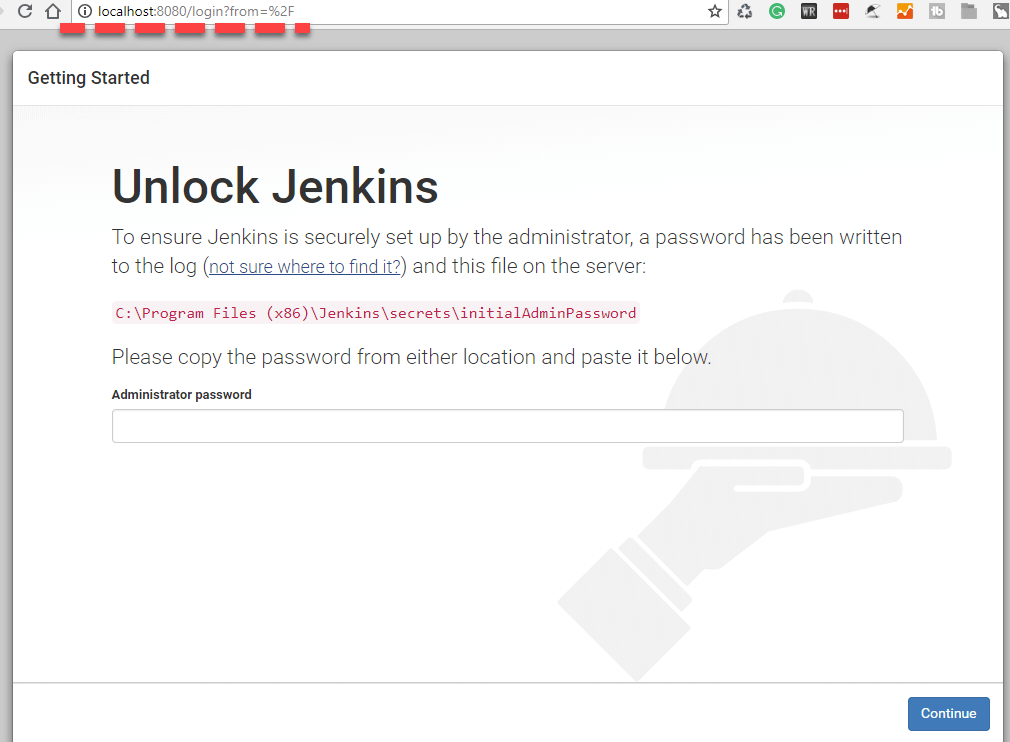
## How to Unblock Jenkins?

After completing the Jenkins installation phase, you should proceed further and start its configuration. Next steps will guide you how you can unblock Jenkins application:

**Step 1)** After completing the Jenkins installation process, a browser tab will pop-up asking for the initial Administrator password. To access Jenkins, you need to go to browse the following path in your web browser.

http://localhost:8080

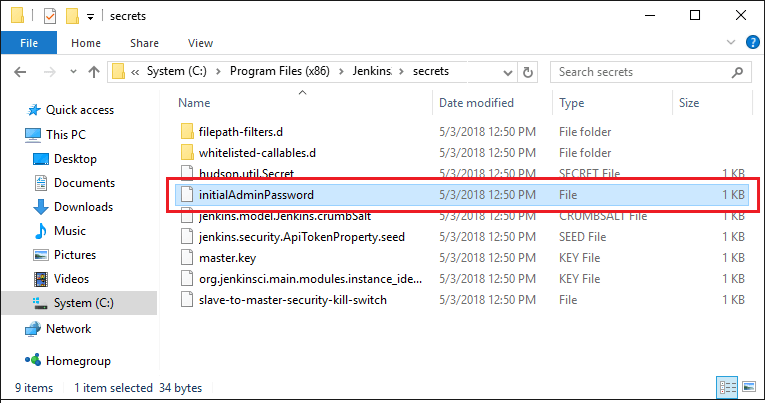
If you can access the above URL, then it confirms that Jenkins is successfully installed in your system.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo8.png)

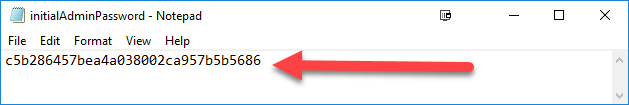
**Step 2)** The initial Administrator password should be found under the Jenkins installation path (set at Step 4 in Jenkins Installation).

For default installation location to C:\Program Files (x86)\Jenkins, a file called **initialAdminPassword** can be found under C:\Program Files (x86)\Jenkins\secrets.

However, If a custom path for Jenkins installation was selected, then you should check that location for **initialAdminPassword** file.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo9.png)

**Step 3)** Open the highlighted file and copy the content of the **initialAdminPassword**file.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo10.png)

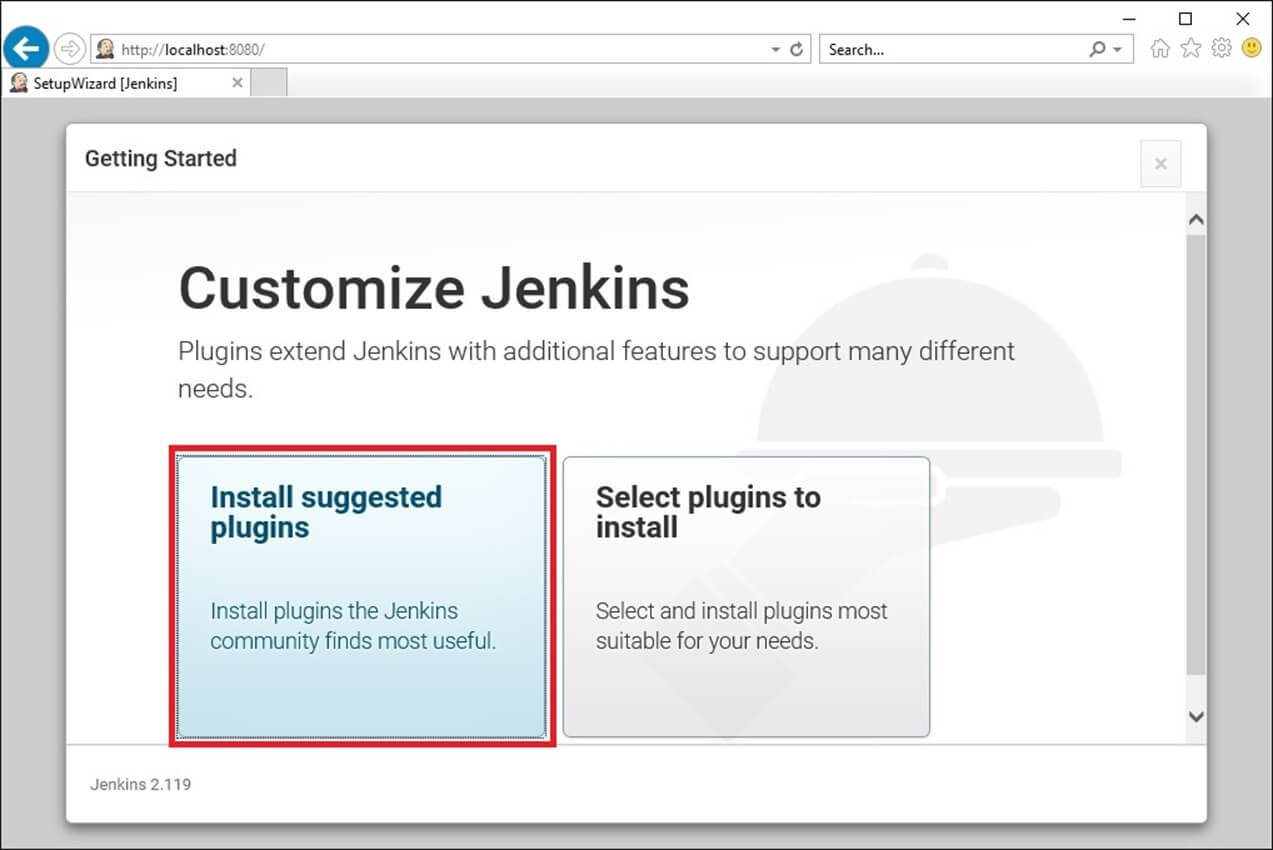
**Step 4)** Paste the password it into browser's pop-up tab (<http://localhost:8080/login?form=%2F>) and click on Continue button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo11.jpg)

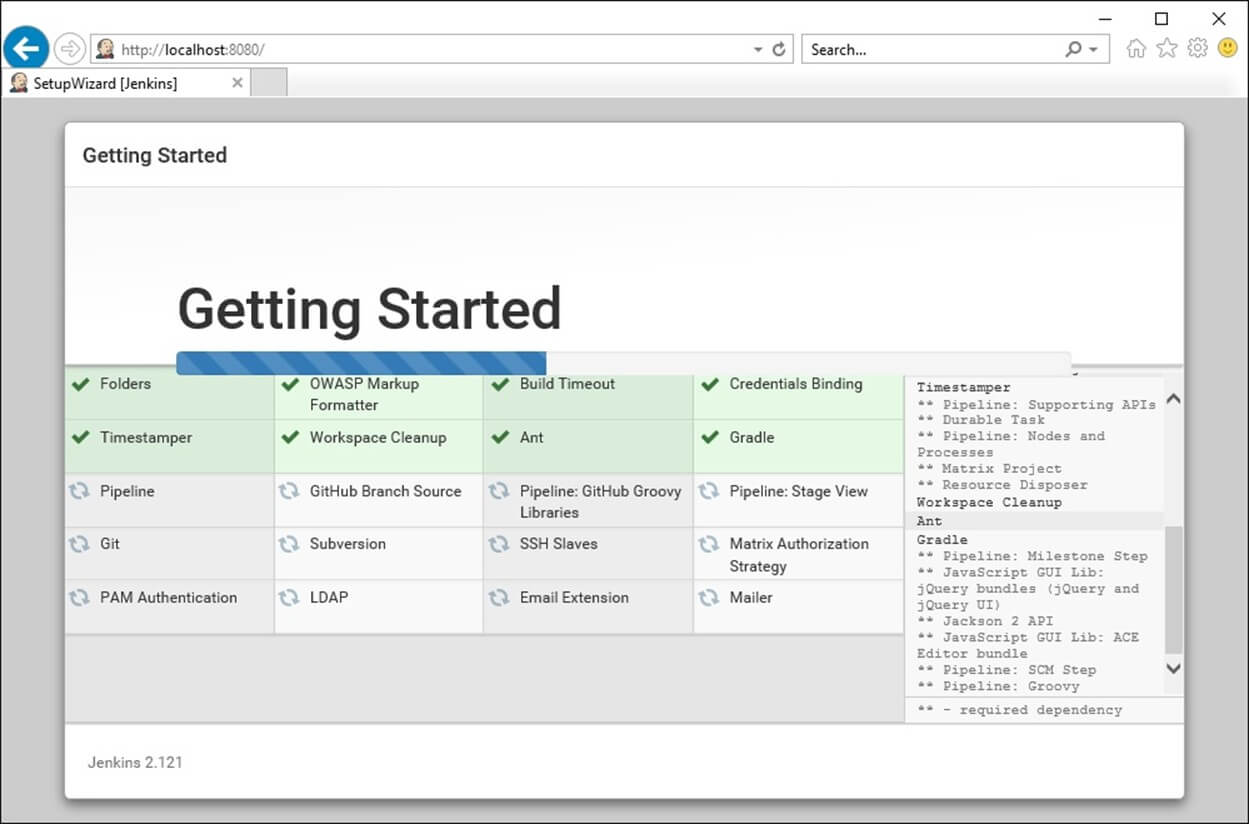
## Customize Jenkins

You can also customize your Jenkins environment by below-given steps:

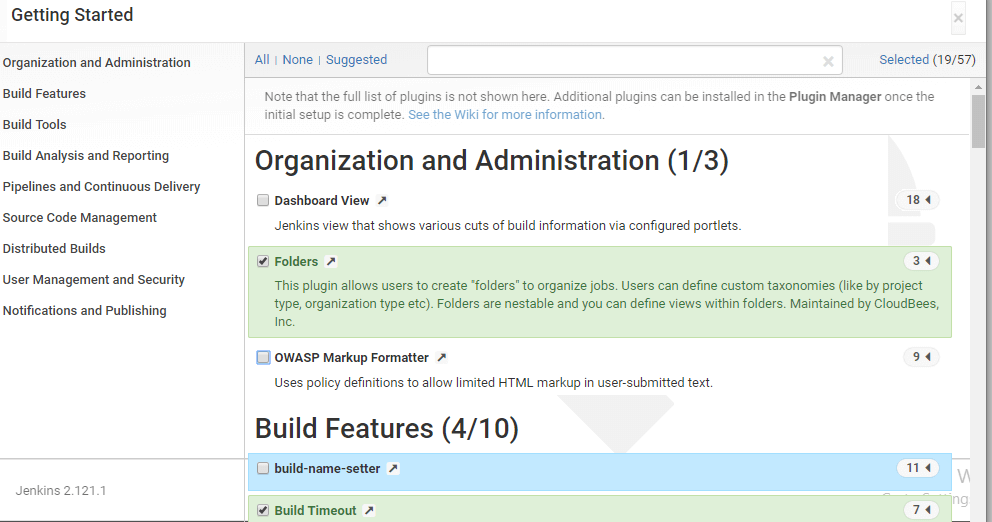
**Step 1)** Click on the "Install suggested plugins button" so Jenkins will retrieve and install the essential plugins

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo12.jpg)

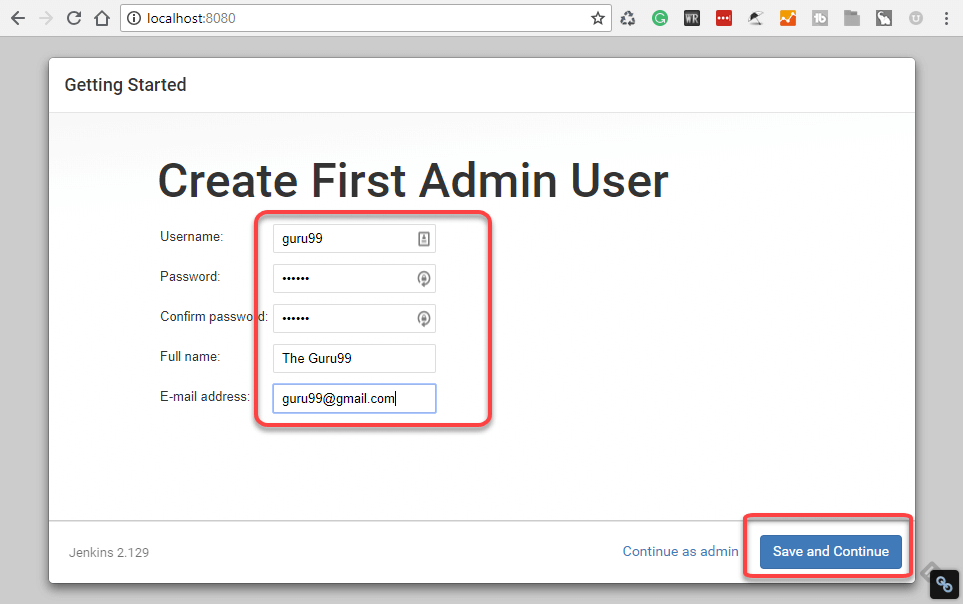
Jenkins will start to download and install all the necessary plugins needed to create new Jenkins Jobs.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo13.jpg)

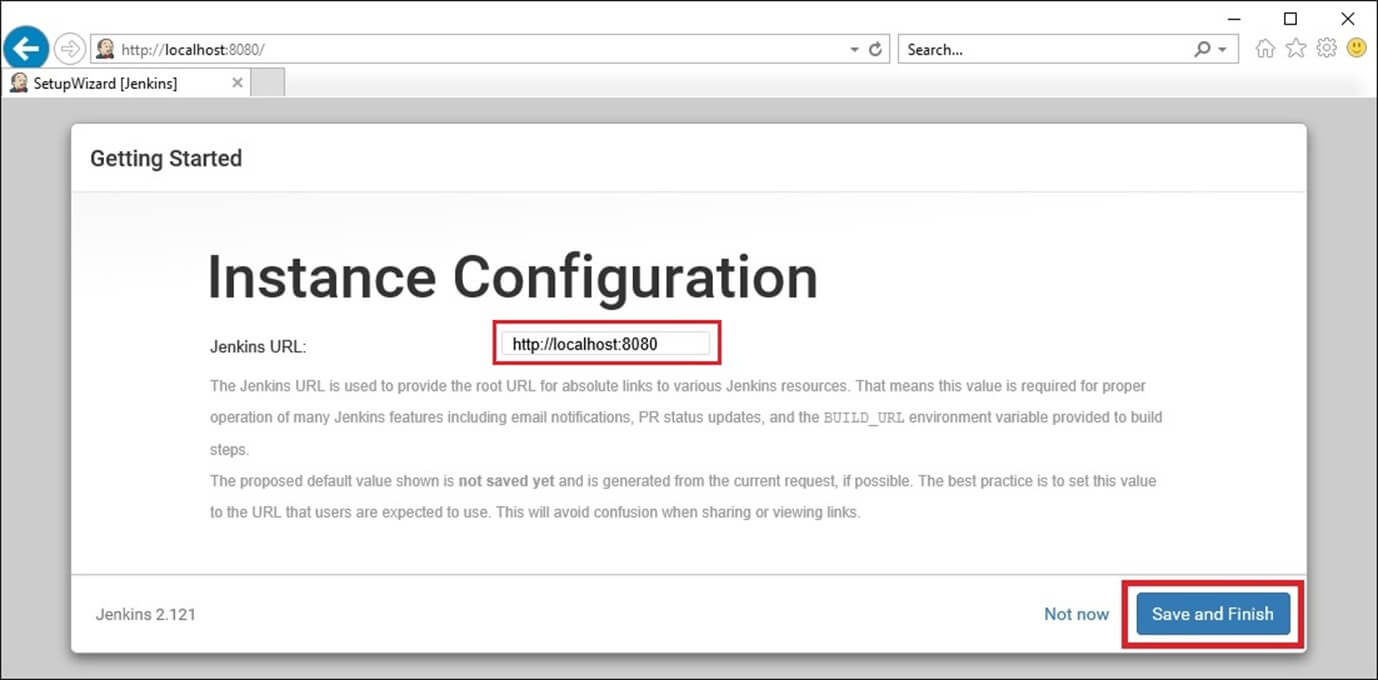
**Note**: You can choose the Option "Select Plugins to Install" and select the plugins you want to install

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo14.png)

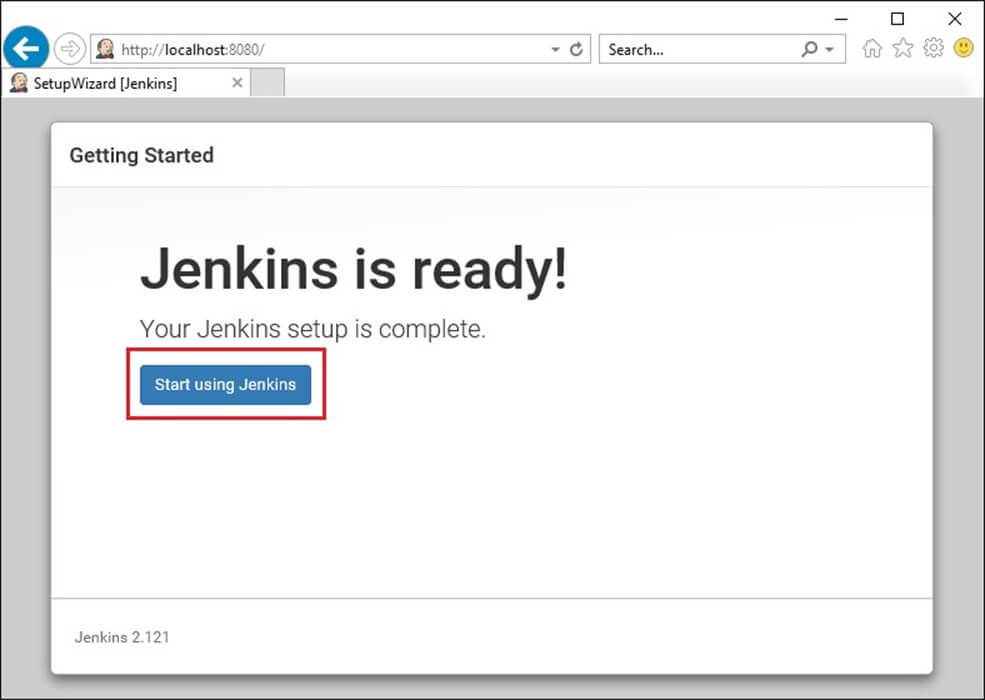
**Step 2)** After all suggested plugins were installed, the "Create First Admin User" panel will show up. Fill all the fields with desired account details and hit the "**Save and Finish**" button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo15.png)

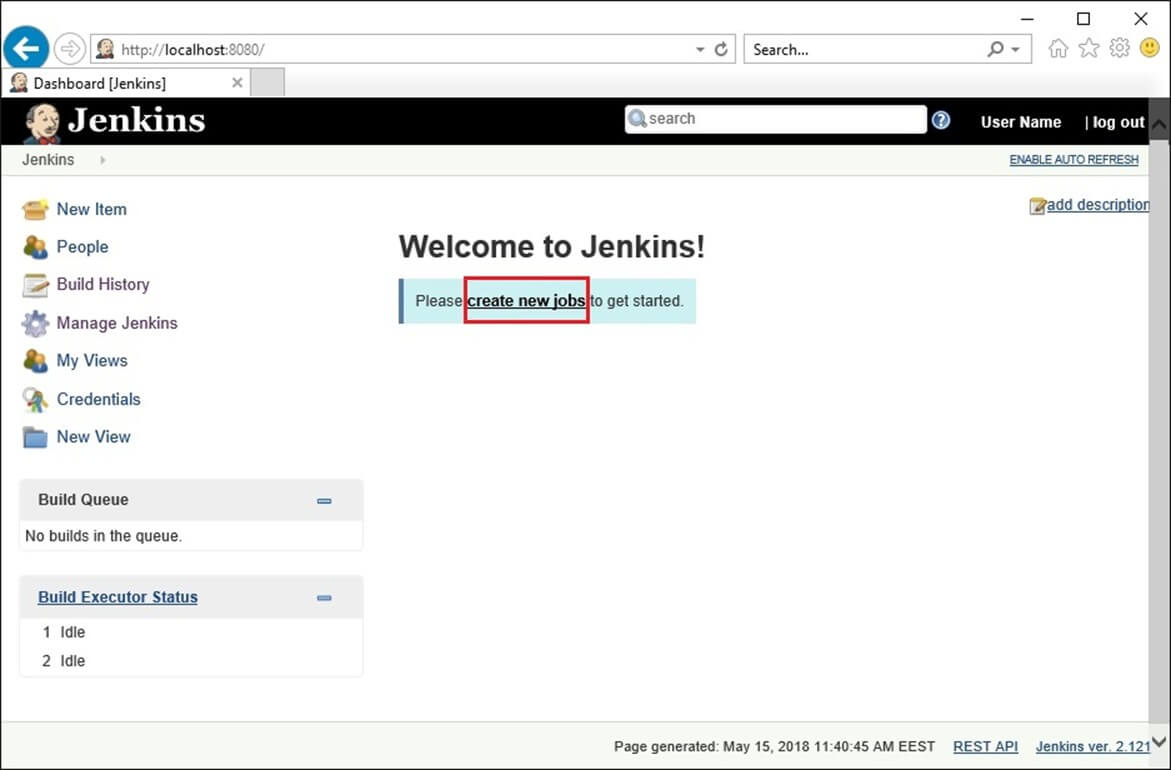
**Step 3)**Once you have filled the above data, finally it will ask for URL information where you can configure the default instance path for Jenkins. Leave it as it is to avoid any confusions later. However, if another application is already using 8080 port, you can use another port for Jenkins and finally save the settings, and you are done with installation of Jenkins. Hit the "**Save and Continue**" button:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo16.jpg)

Congratulations! We have successfully installed a new Jenkins Server. Hit the "Start using Jenkins" button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo17.jpg)

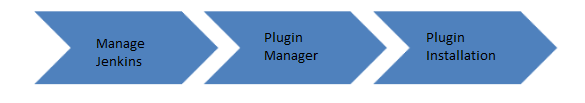
Below you can find the Jenkins instance up and run, ready to create first Jenkins jobs:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo18.jpg)

# Jenkins GitHub Integration: Install Git Plugin

You have learned, from previous Guru99 tutorials, what Jenkins is and how to install it onto a Windows system. Assuming you have completed those basic steps, we shall now move on to Plugin management.

Jenkins has outstanding plugin support. There are thousands of third-party application plugins available on their website. To know if Jenkins supports the third-party applications you have in mind, check their plugins directory at <https://wiki.jenkins-ci.org/display/JENKINS/Plugins>.

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH1.png)

In this tutorial, you will learn:

* [Installation of Plugins in Jenkins](https://www.guru99.com/jenkins-github-integration.html#1)
* [Install GIT Plugin](https://www.guru99.com/jenkins-github-integration.html#2)
* [Integrating Jenkins with GitHub](https://www.guru99.com/jenkins-github-integration.html#3)

## Installation of Plugins in Jenkins

Jenkins comes with a pretty basic setup, so you will need to install the required plugins to enable respective third-party application support.

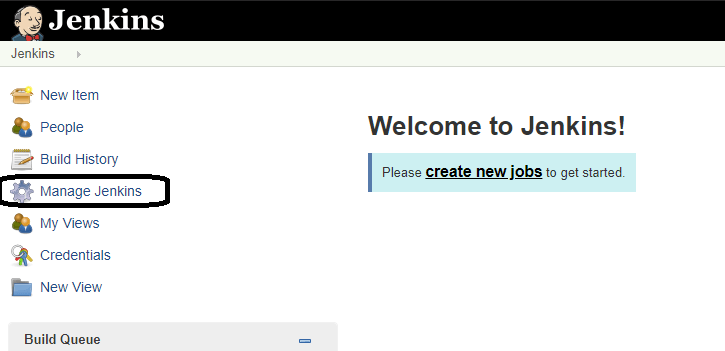
GitHub is a web-based repository of code which plays a major role in DevOps. It provides a common platform for multiple developers working on the same code/project to upload and retrieve updated code, thereby facilitating continuous integration.

Jenkins needs to have GitHub plugin installed to be able to pull code from the GitHub repository.

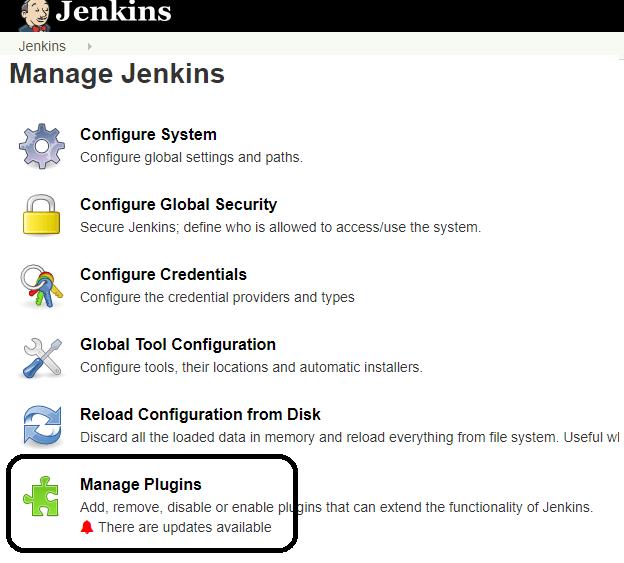
You need not install a GitHub plugin if you have already installed the Git plugin in response to the prompt during the Jenkins' installation setup. But if not, here is how you install GitHub plugins in Jenkins and pull code from a GitHub repository.

## Install GIT Plugin

**Step 1:**Click on the **Manage Jenkins** button on your Jenkins dashboard:

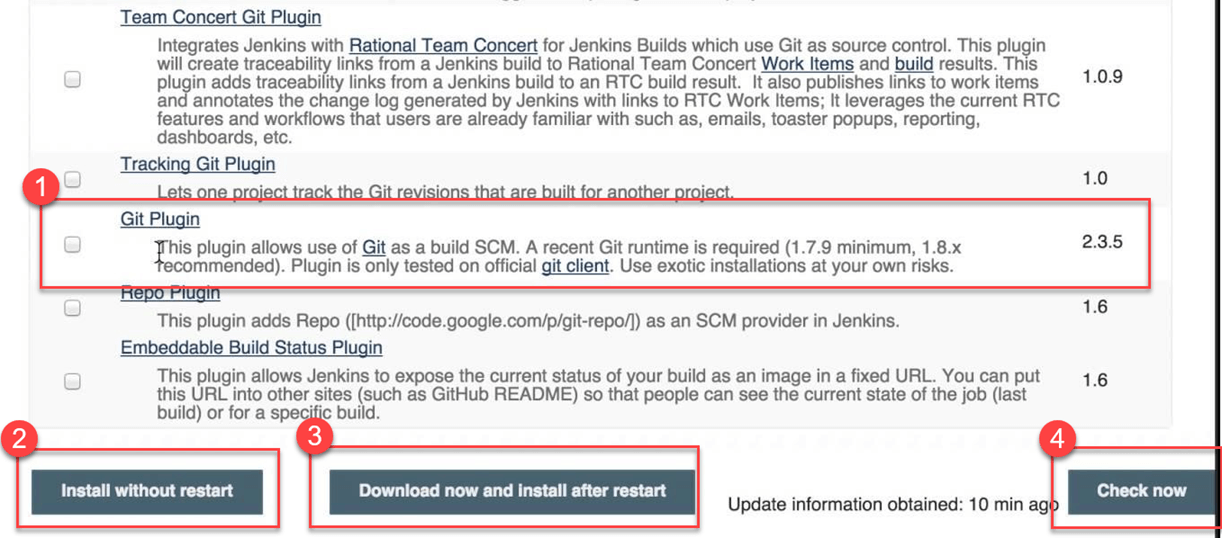
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH2.png)

**Step 2:**Click on **Manage Plugins**:

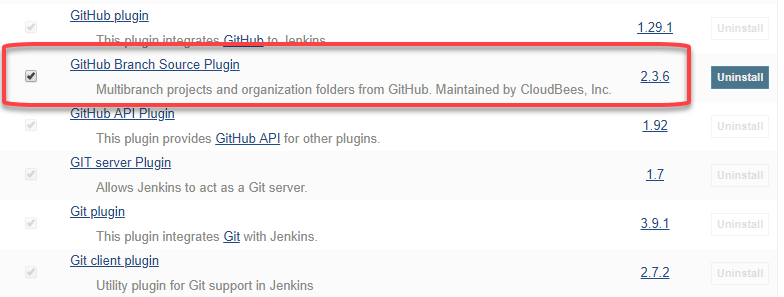
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH3.png)

**Step 3:**In the Plugins Page

1. Select the GIT Plugin
2. Click on **Install without restart.**The plugin will take a few moments to finish downloading depending on your internet connection, and will be installed automatically.
3. You can also select the option **Download now and Install after restart**button. In which plugin is installed after restart
4. You will be shown a "No updates available" message if you already have the Git plugin installed.

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH4.png)

**Step 4:** Once the plugins have been installed, go to **Manage Jenkins**on your Jenkins dashboard. You will see your plugins listed among the rest.

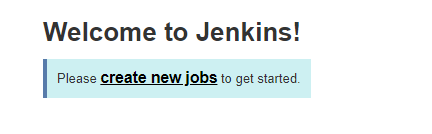
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH5.png)

## Integrating Jenkins with GitHub

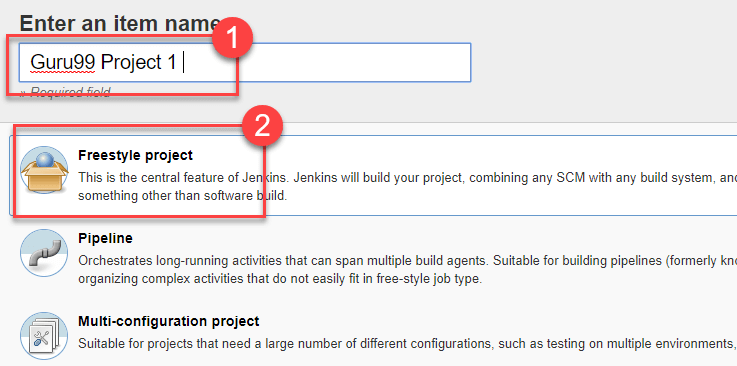
We shall now discuss the process of integrating GitHub into Jenkins in a Windows system.

**Step 1)**Create a new job in Jenkins, open the Jenkins dashboard with your Jenkins URL. For example, <http://localhost:8080/>

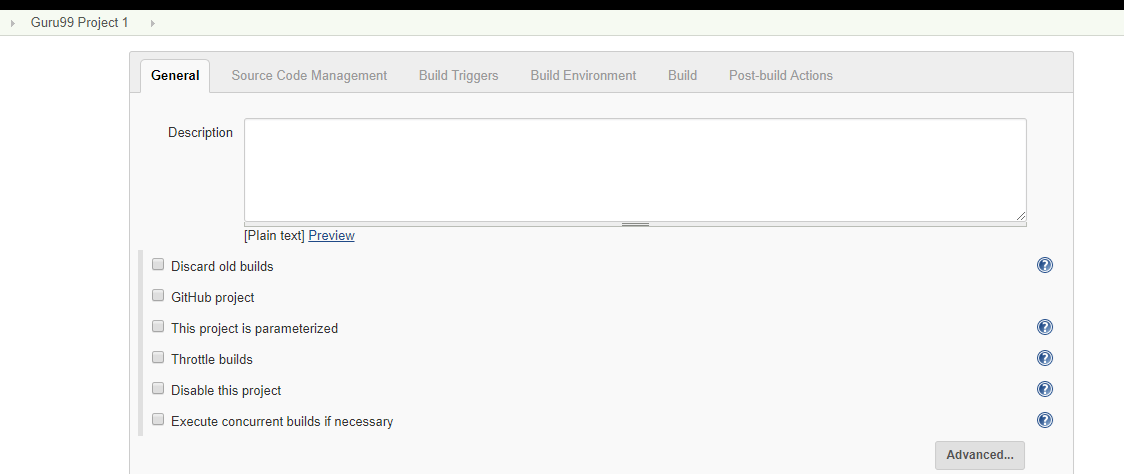
Click on **create new jobs**:

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH6.png)

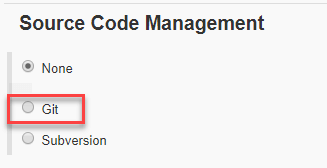
**Step 2)**Enter the item name, select job type and click **OK**. We shall create a Freestyle project as an example.

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH7.png)

**Step 3)**Once you click **OK,**the page will be redirected to its project form. Here you will need to enter the project information:

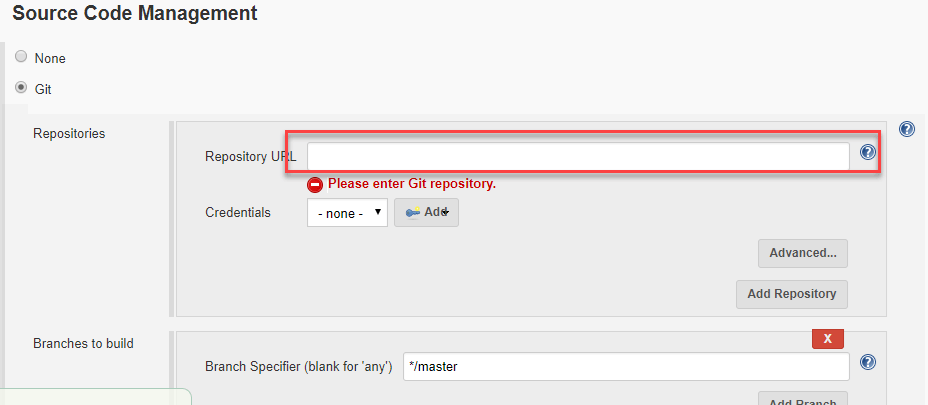
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH8.png)

**Step 4)**You will see a **Git**option under **Source Code Management**if your Git plugin has been installed in Jenkins:

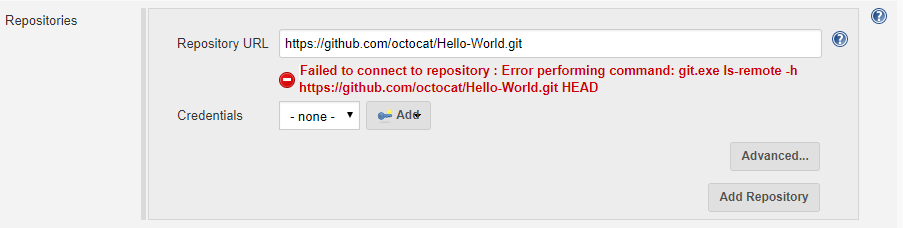
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH9.png)

NOTE: If the **Git**option does not appear, try re-installing the plugins, followed by a restart and a re-login into your Jenkins dashboard.You will now be able to see the **Git**option as mentioned above.

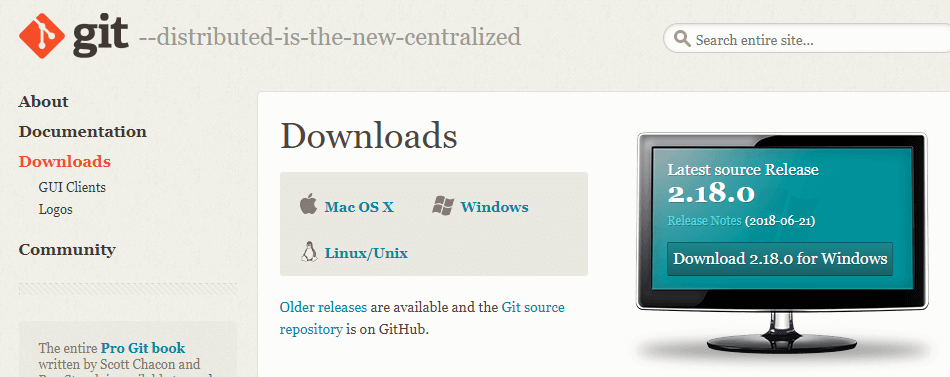
**Step 5)**Enter the Git repository URL to pull the code from GitHub.

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH10.png)

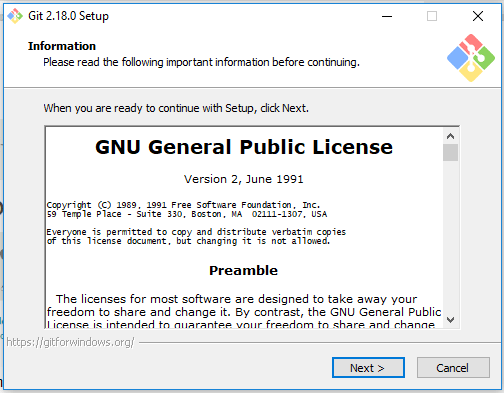
**Step 6)**You might get an error message the first time you enter the repository URL. For example:

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH11.png)

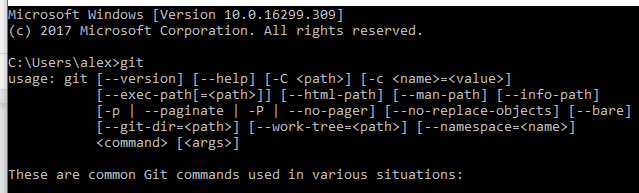
This happens if you do not have Gitinstalled in your local machine. To install Git in your local machine, go to <https://git-scm.com/downloads>

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH12.png)

Download the appropriate Git file for your Operating System, in this case, Windows, and install it onto your local machine running Jenkins. Complete the onscreen instructions to install GIT.

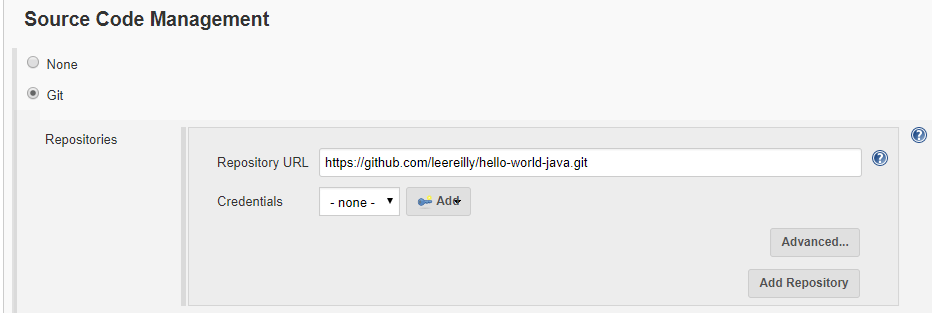
[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH13.png)

**Step 6)**You can execute Git repositories in your Jenkins once Git has been installed on your machine. To check ifithas been successfully installed onto your system, open your **command prompt,** type "Git"and press enter. You should see different options come up for Git:

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH14.png)

This means that Git has been installed in your system.

**Step 7)**Once you have everything in place, try adding the Git URL into Jenkins. You will not see any error messages:

[](https://www.guru99.com/images/1/091318_0440_JenkinsGitH15.png)

Git is now properly configured on your system.

# How to Create Users & Manage Permissions: Role Strategy Plugin

Generally, in a large organization, there are multiple, separate teams to manage and run jobs in Jenkins. But managing this crowd of users and assigning roles to them can prove troublesome.

By default, Jenkins comes with very basic user creation options. You can create multiple users but can only assign the same roles and privileges to them. This not ideal, especially for a large organization.

The **Role Strategy Plugin**allows you to assign different roles and privileges to different users**.** You will first need to install the plugin in your Jenkins environment.

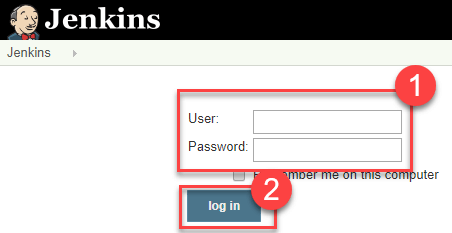
In this tutorial, you will learn

* [How to Create/Add a User](https://www.guru99.com/create-users-manage-permissions.html#1)
* [Install Role Strategy Plugin](https://www.guru99.com/create-users-manage-permissions.html#2)
* [Create Roles](https://www.guru99.com/create-users-manage-permissions.html#3)
* [Assign a Role](https://www.guru99.com/create-users-manage-permissions.html#4)
* [Project Roles](https://www.guru99.com/create-users-manage-permissions.html#5)

## How to Create/Add a User

**Step 1)**Login to your Jenkins dashboard by visiting <http://localhost:8080/>

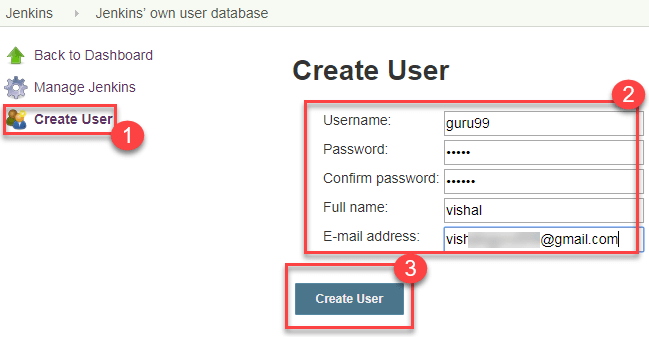
If you haven't installed Jenkins in your local server, go to the appropriate URL and access your dashboard by using your login credentials.

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate1.png)

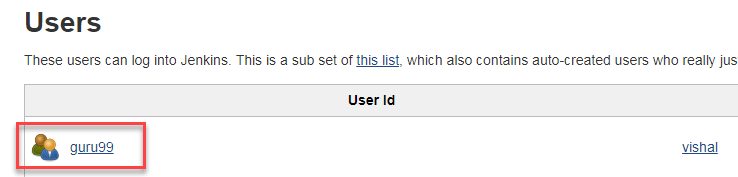
**Step 2)**You will now see options to create new users and manage current users.

**Step 3)**

* Under Manage Jenkins, Click Create User
* Enter User details like password, name, email etc.
* Click Create User

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate2.png)

**Step 4)**User is created

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate3.png)

## Install Role Strategy Plugin

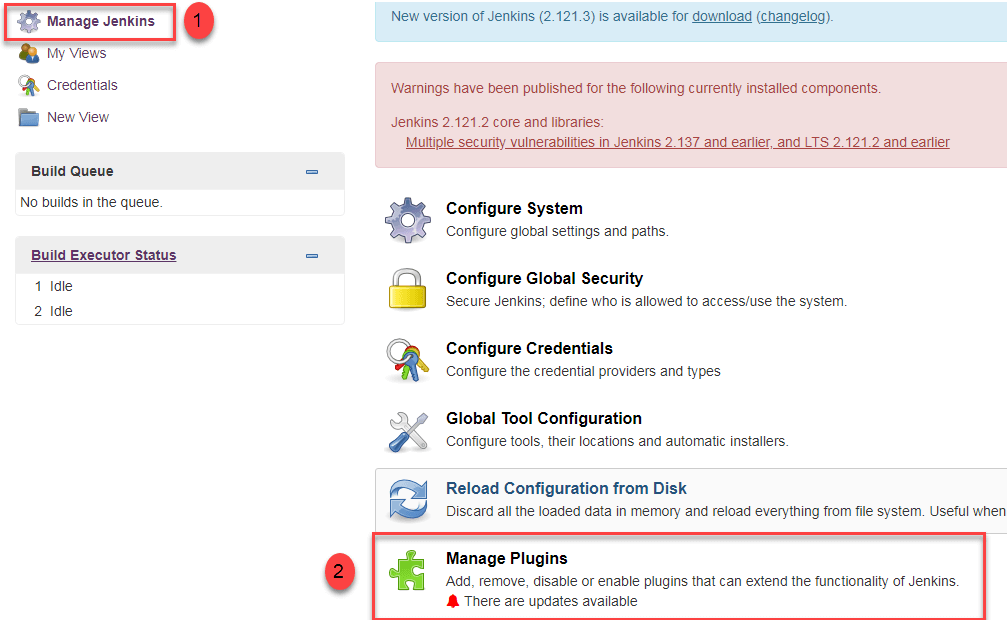
There are two methods for installing plugins in Jenkins:

1. Installing it through your Jenkins dashboard
2. Downloading the plugin from Jenkins website and installing it manually.

**Step 1)**

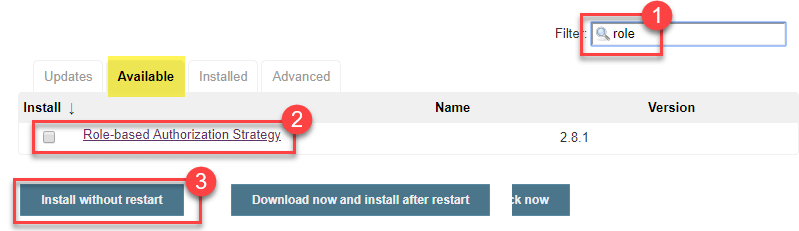
1. Go to **Manage Jenkins**

2. Click on the Manage Plugins option

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate4.png)

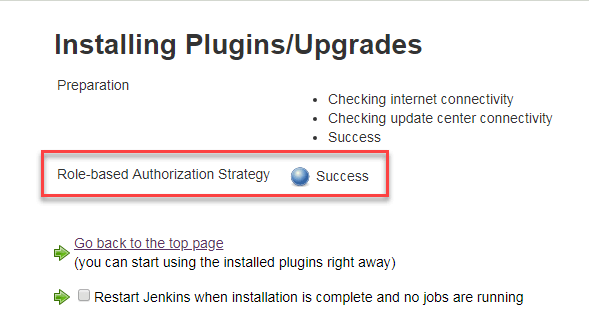
**Step 2)**

1. In available section, screenSearch for "role".
2. Select Role**-based Authorization Strategy**plugin
3. Click on "**Install without restart**" (make sure you have an active internet connection)

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate5.png)

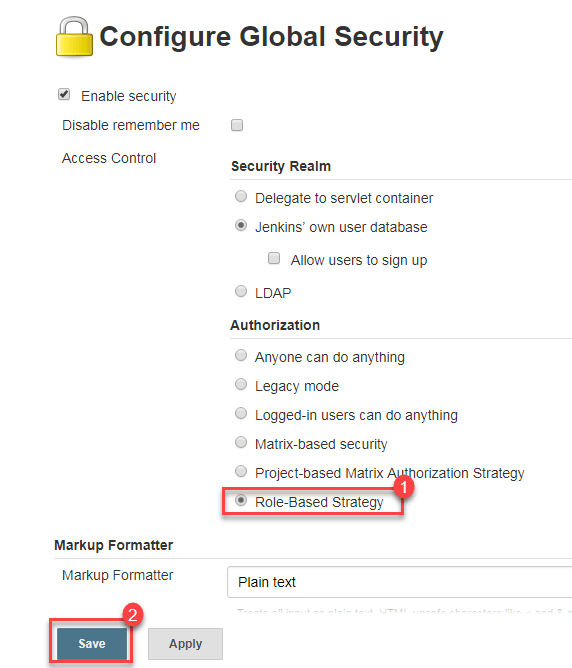
**Step 3)**

Once the plugin is installed, a "success" status will be displayed.

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate6.png)

Click on **Go back to the top page.**

**Step 4)**Go to **Manage Jenkins ->**Configure Global Security -> Under **Authorization,**select **Role Based Strategy**.Click on**Save**.

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate7.png)

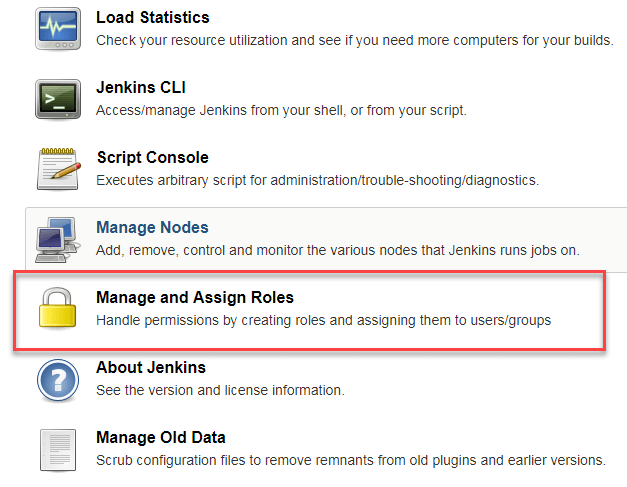
## Create Roles

In this step, we shall learn to certain roles to a single user or a group of users.

**Step 1)**

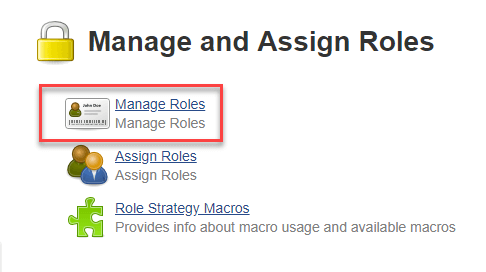
1. Go to **Manage Jenkins**

2. Select **Manage and Assign Roles**

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate8.png)

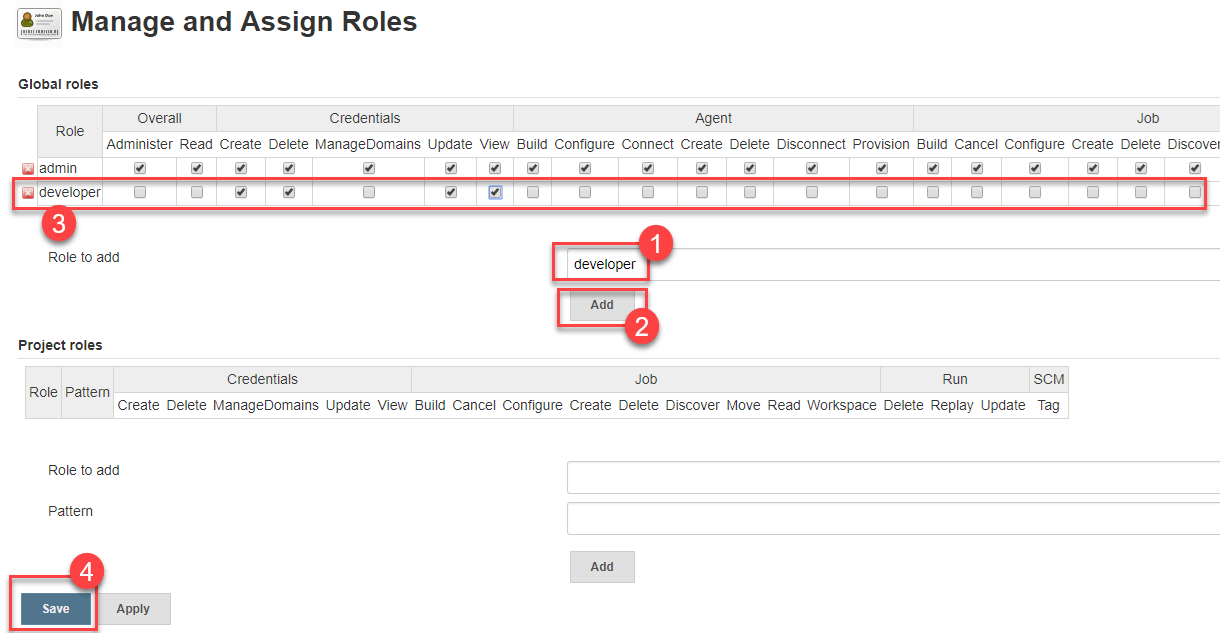
**Note:**that the **Manage and Assign Roles**option will only be visible if you've installed the role strategy plugin.

**Step 2)**Click on **Manage Roles**to add new roles based on your organization.

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate9.png)

**Step 3)**To create a new role called "developer",

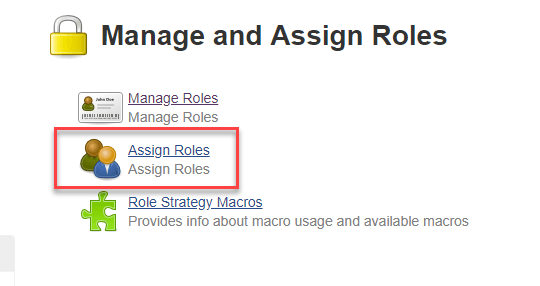
1. Type "developer" under "role".
2. Click on "Add" to create a new role.
3. Now, select the permissions you want to assign to the "Developer" role.
4. Click Save

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate10.png)

## Assign a Role

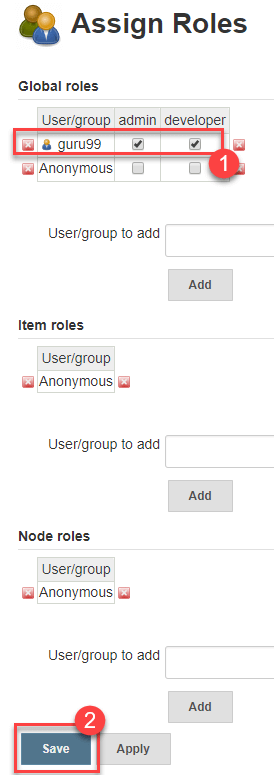
**Step 1)**Now that you have created roles, let us assign them to specific users.

1. Go to **Manage Jenkins**
2. Select Manage and Assign Roles

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate11.png)

**Step 2)**We shall add the new role "developer" to user "**guru99**"

1. Selector developer role checkbox
2. Click Save

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate12.png)

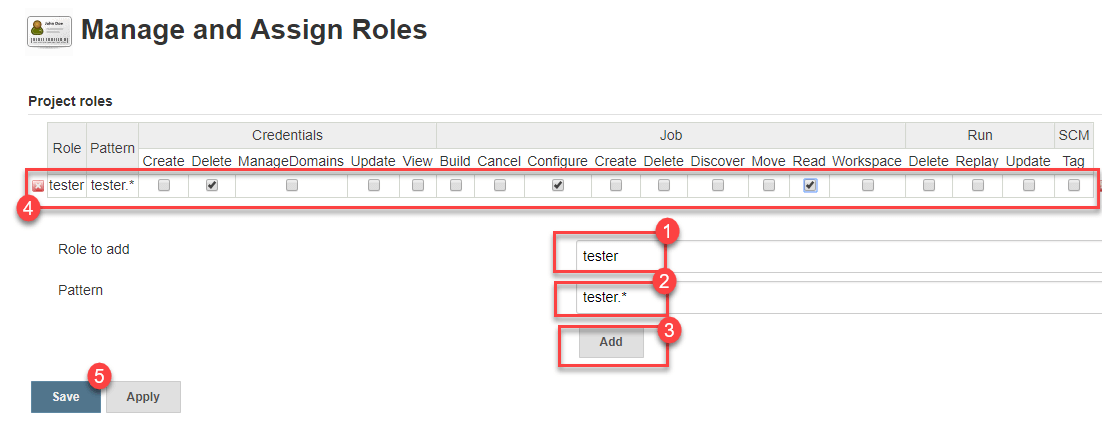
You can assign any role to any user, as per your need.

## Project Roles

You can create project specific roles under **Project Roles.**

**Step 1)**In Manage and Assign Roles

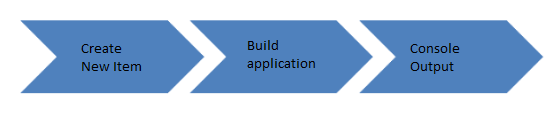
1. Enter a role as "tester"
2. Add a pattern to this by adding **tester.\***, so that any username starting with "tester" will be assigned the project roles you specify.
3. Click Add
4. Select privileges
5. Click Save

[](https://www.guru99.com/images/1/091318_0444_HowtoCreate13.png)

# How to Create Builds with the Jenkins Freestyle Project

## What is a Jenkins freestyle project?

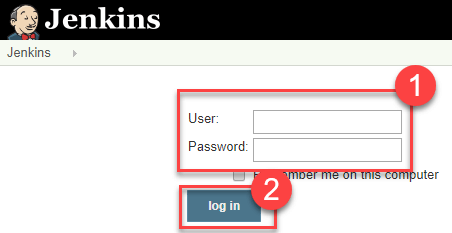
A Jenkins project is a repeatable build job which contains steps and post-build actions. The types of actions you can perform in a build step or post-build action are quite limited. There are many standard plugins available within a Jenkins freestyle project to help you overcome this problem. They allow you to configure build triggers and offers project-based security for your Jenkins project.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate1.png)

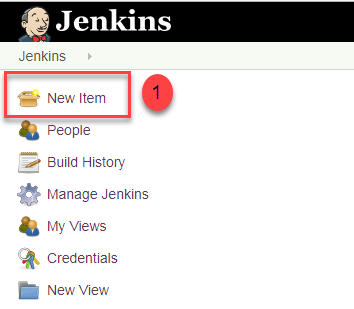
## Creating a Freestyle Build Job

The freestyle build job is a highly flexible and easy-to-use option. You can use it for any type of project; it is easy to set up, and many of its options appear in other build jobs.

**Step 1)**To create a Jenkins freestyle job, log on to your Jenkins dashboard by visiting your Jenkins installation path. Usually, it will be hosted on localhost at [http://localhost:8080](http://localhost:8080/) If you have installed Jenkins in another path, use the appropriate URL to access your dashboard.

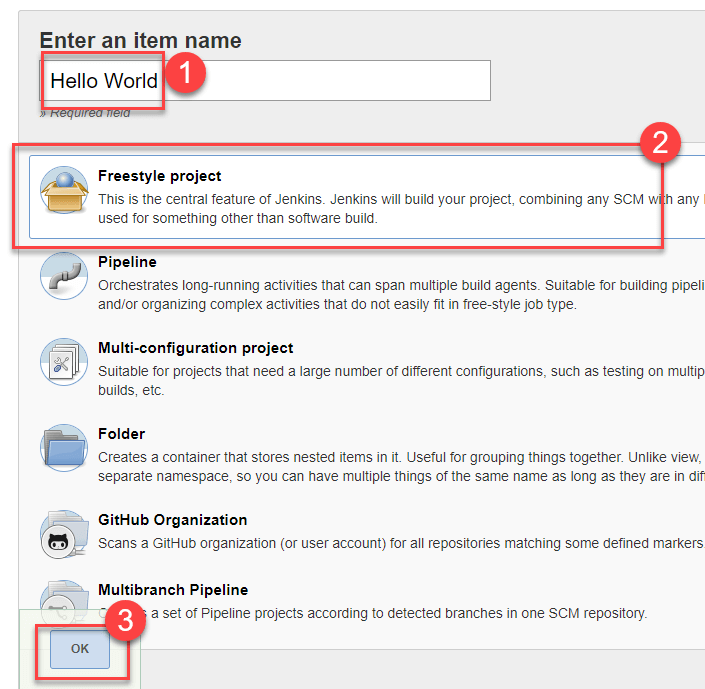
[](https://www.guru99.com/images/1/091318_0458_HowtoCreate2.png)

**Step 2)**Click on "**New Item**" at the top left-hand side of your dashboard.

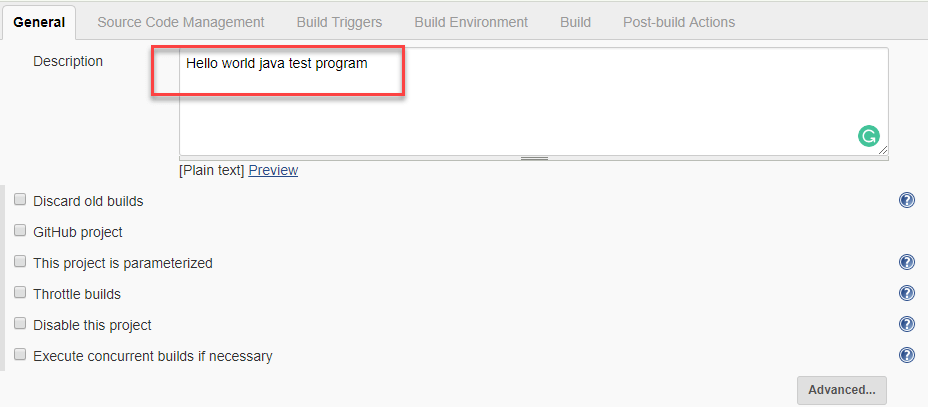
[](https://www.guru99.com/images/1/091318_0458_HowtoCreate3.png)

**Step 3)**In the next screen,

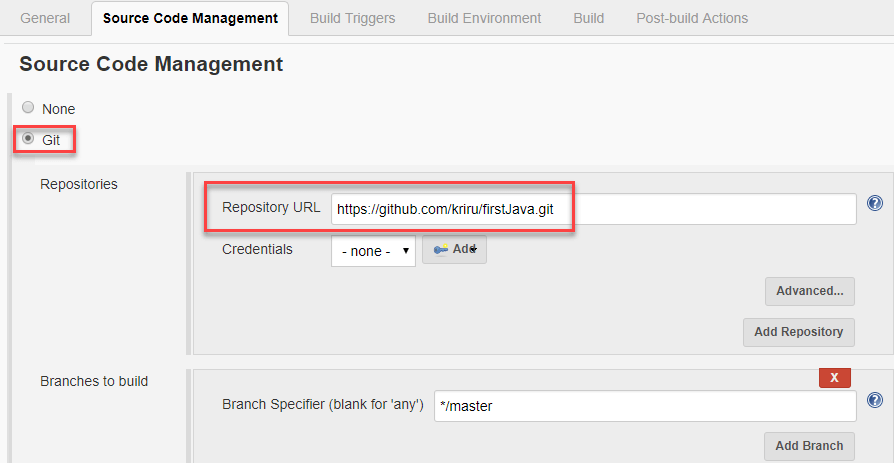
1. Enter the name of the item you want to create. We shall use the "Hello world" for this demo.
2. Select Freestyle project
3. Click Okay

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate4.png)

**Step 4)**Enter the details of the project you want to test.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate5.png)

**Step 5)**Under Source Code Management, Enter your repository URL. We have a test repository located at <https://github.com/kriru/firstJava.git>

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate6.png)

It is also possible for you to use a local repository.

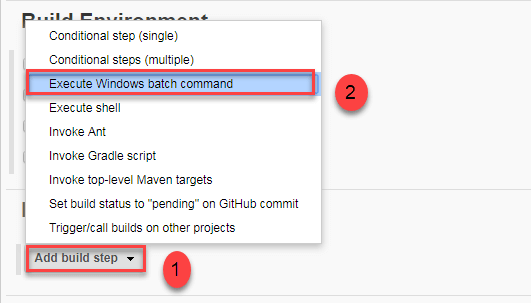
If your GitHub repository is private, Jenkins will first validate your login credentials with GitHub and only then pull the source code from your GitHub repository.

**Step 6)**Now that you have provided all the details, it's time to build the code. Tweak the settings under the **build** section to build the code at the time you want. You can even schedule the build to happen periodically, at set times.

Under **build**,

1. Click on "**Add build step**"

2. Click on "**Execute Windows batch command**" and add the commands you want to execute during the build process.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate7.png)

Here, I have added the java commands to compile the java code.

I have added the following windows commands:

javac HelloWorld.java

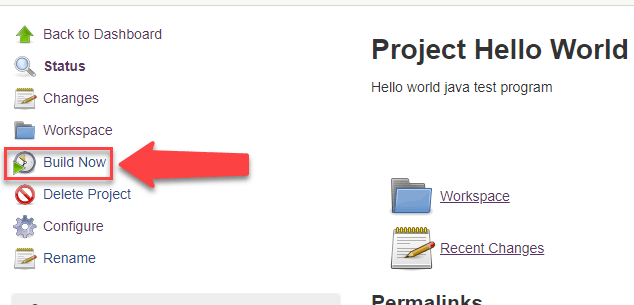
java HelloWorld

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate8.png)

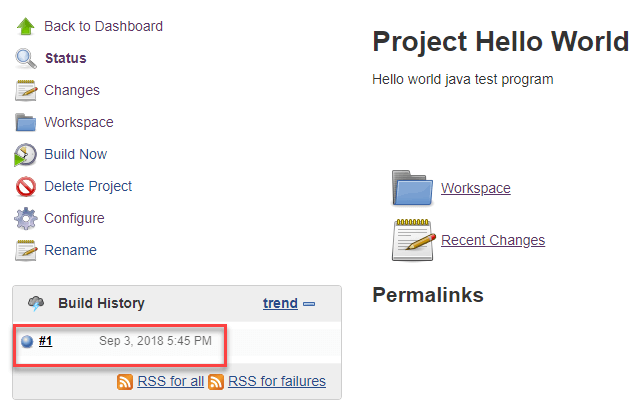
**Step 7)** When you have entered all the data,

1. Click **Apply**
2. **Save**the project.

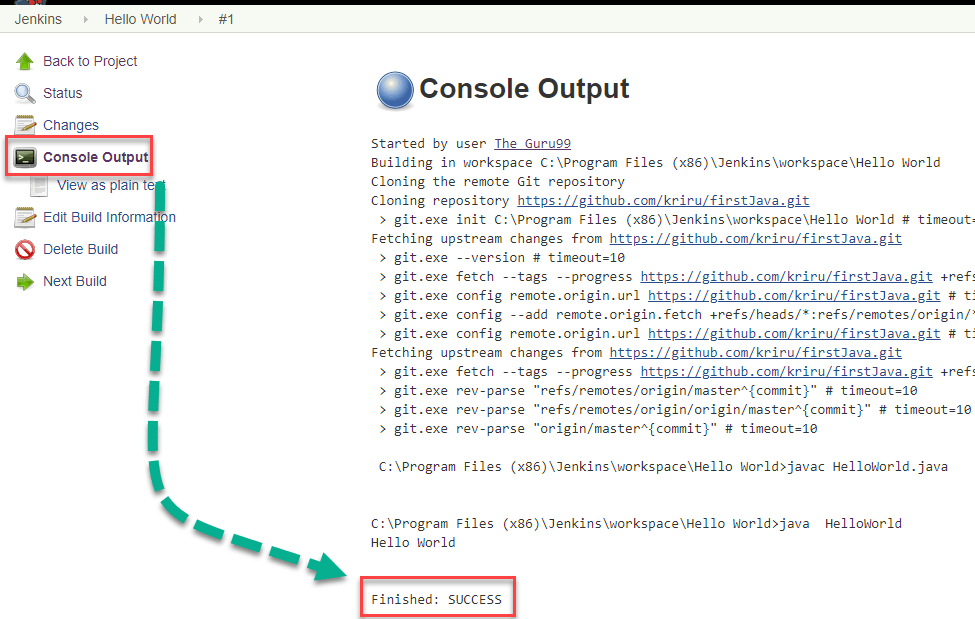
**Step 8)**Now, in the main screen, Click the **Build Now**button on the left-hand side to build the source code.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate9.png)

**Step 9)**After clicking on **Build now,**you can see the status of the build you run under **Build History**.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate10.png)

**Step 10)**Click on the **build number**andthenClick on **console output**to see the status of the build you run. It should show you a success message, provided you have followed the setup properly.

[](https://www.guru99.com/images/1/091318_0458_HowtoCreate11.png)

In sum, we have executed a HelloWorld program hosted on GitHub. Jenkin pulls the code from the remote repository and builds continuously at a frequency you define.

# Jenkins Pipeline Tutorial: JenkinsFile EXAMPLE

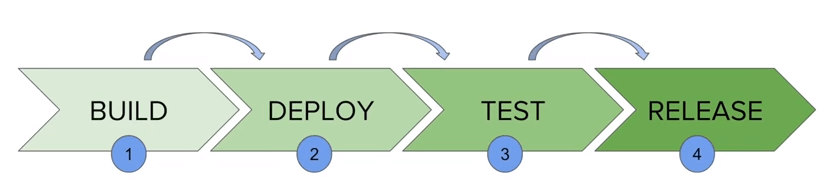
## What is Jenkins Pipeline?

In Jenkins, a pipeline is a group of events or jobs which are interlinked with one another in a sequence.

In simple words, Jenkins Pipeline is a combination of plugins that support the integration and implementation of **continuous delivery pipelines** using Jenkins. A pipeline has an extensible automation server for creating simple or complex delivery pipelines "as code," via pipeline DSL (Domain-specific Language).

### What is Continuous Delivery Pipelines? How it Works?

In a Jenkins pipeline, every job or event has some sort of dependency on at least one or more events.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe1.png)

The picture above represents a continuous delivery pipeline in Jenkins. It contains a group of states called build, deploy, test and release. These events are interlinked with each other. Every state has its events, which work in a sequence called a continuous delivery pipeline.

A continuous delivery pipeline is an automated expression to display your process for getting software for version control. Thus, every change made in your software goes through a number of complex processes on its way to being released. It also involves developing the software in a reliable and repeatable manner, and progression of the built software through multiple stages of testing and deployment.

In this tutorial, you will learn

* [What is Jenkins Pipeline?](https://www.guru99.com/jenkins-pipeline-tutorial.html#1)
* [What is a JenkinsFile?](https://www.guru99.com/jenkins-pipeline-tutorial.html#2)
* [Why Use Jenkin's Pipeline?](https://www.guru99.com/jenkins-pipeline-tutorial.html#3)
* [Jenkins Pipeline Concepts](https://www.guru99.com/jenkins-pipeline-tutorial.html#4)
* [Install Build Pipeline Plugin in Jenkins](https://www.guru99.com/jenkins-pipeline-tutorial.html#5)
* [How to Create Jenkins Pipeline](https://www.guru99.com/jenkins-pipeline-tutorial.html#6)
* [Running Jenkins pipeline](https://www.guru99.com/jenkins-pipeline-tutorial.html#7)
* [Best Practices using Jenkins Pipeline](https://www.guru99.com/jenkins-pipeline-tutorial.html#8)

## What is a JenkinsFile?

Jenkins pipelines can be defined using a text file called **JenkinsFile.**You can implement pipeline as code using JenkinsFile, and this can be defined by using a domain specific language (DSL). With JenkinsFile, you can write the steps needed for running a Jenkins pipeline.

The benefits of using J**enkinsFile are**:

* You can create pipelines automatically for all branches and execute pull requests with just one **JenkinsFile.**
* You can review your code on the pipeline
* You can audit your Jenkins pipeline
* This is the singular source for your pipeline and can be modified by multiple users.

JenkinsFile can be defined by either Web UI or with a JenkinsFile.

**Declarative versus Scripted pipeline syntax:**

There are two types of syntax used for defining your JenkinsFile.

1. Declarative
2. Scripted

**Declarative:**

Declarative pipeline syntax offers an easy way to create pipelines. It contains a predefined hierarchy to create Jenkins pipelines. It gives you the ability to control all aspects of a pipeline execution in a simple, straight-forward manner.

**Scripted:**

Scripted Jenkins pipeline runs on the Jenkins master with the help of a lightweight executor. It uses very few resources to translate the pipeline into atomic commands. Both declarative and scripted syntax are different from each other and are defined totally differently.

## Why Use Jenkin's Pipeline?

Jenkins is an open continuous integration server which has the ability to support the automation of software development processes. You can create multiple automation jobs with the help of use cases, and run them as a Jenkins pipeline.

Here are the reasons why you use should use Jenkins pipeline:

* Jenkins pipeline is implemented as a code which allows multiple users to edit and execute the pipeline process.
* Pipelines are robust. So if your server undergoes an unforeseen restart, the pipeline will be automatically resumed.
* You can pause the pipeline process and make it wait to resume until there is an input from the user.
* Jenkins Pipelines support big projects. You can run multiple jobs, and even use pipelines in a loop.

## Jenkins Pipeline Concepts

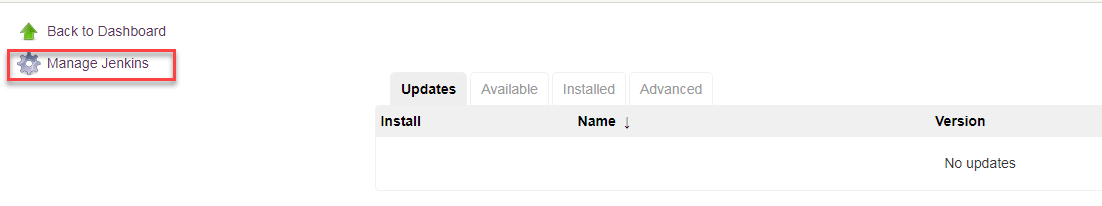
|  |  |
| --- | --- |
| **Term** | **Description** |
| Pipeline | The pipeline is a set of instructions given in the form of code for continuous delivery and consists of instructions needed for the entire build process. With pipeline, you can build, test, and deliver the application. |
| Node | The machine on which Jenkins runs is called a node. A node block is mainly used in scripted pipeline syntax. |
| Stage | A stage block contains a series of steps in a pipeline. That is, the build, test, and deploy processes all come together in a stage. Generally, a stage block is used to visualize the Jenkins pipeline process. |
| Step | A step is nothing but a single task that executes a specific process at a defined time. A pipeline involves a series of steps. |

## Install Build Pipeline Plugin in Jenkins

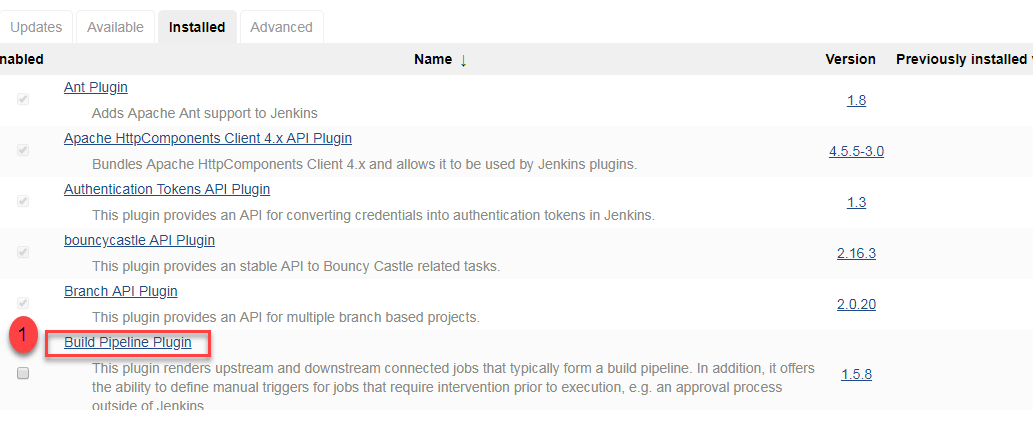
With the **build pipeline** plugin, you can create a pipeline view of incoming and outgoing jobs, and create triggers which require manual intervention.

Here is how you can install the **build pipeline**plugin in your Jenkins:

**Step 1**) The settings for the plugin can be found under **Manage Jenkins > Manage Plugins.**

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe2.png)

If you have already installed the plugin, it is shown under the installed tab.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe3.png)

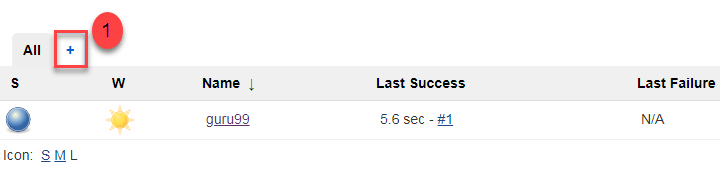
**Step 2**) If you do not have the plugin previously installed, it shows up under the **Available**tab.

Once you have successfully installed the **build pipeline** plugin in your Jenkins, follow these steps to create your Jenkins pipeline:

## How to Create Jenkins Pipeline

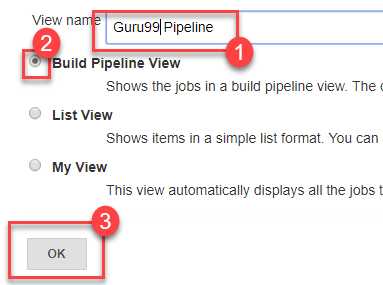
Once you are logged in to your Jenkins dashboard:

**Step 1**) Click on the **"+"**button on the left-hand side of your Jenkins dashboard to create a pipeline.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe4.png)

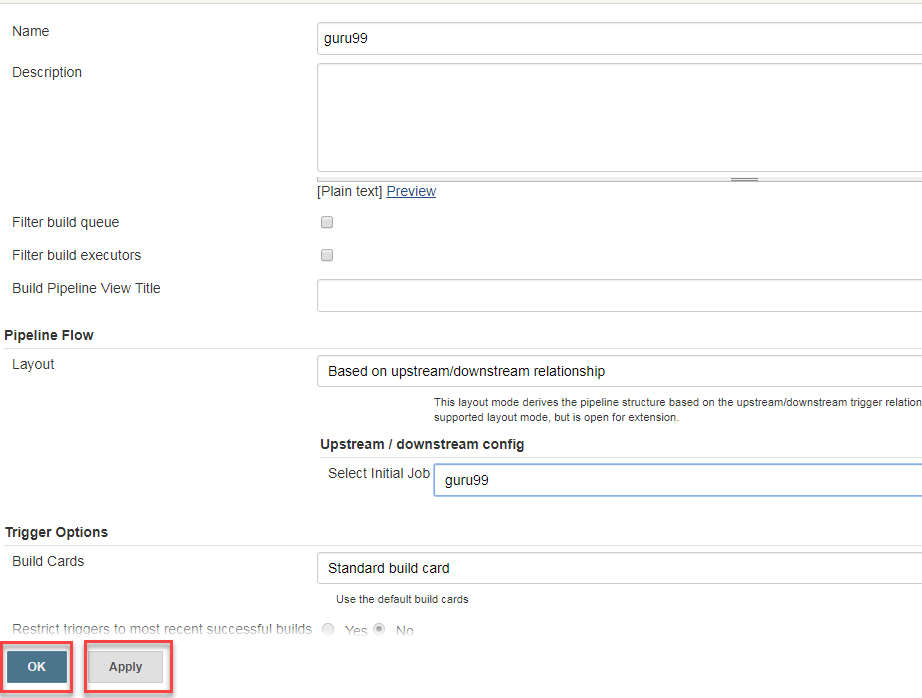
**Step 2**)

1. You will be asked to give a name to the pipeline view. We shall call it "**Guru99 Pipeline**" for the duration of this demo.
2. Select **Build a pipeline view**under **options**
3. Click **ok**

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe5.png)

**Step 4**) In the next page, you will be asked for some more details to configure your Jenkins pipeline. Just accept the default settings, and make sure you choose the first job under the settings.

Click on **Apply**and then **OK**.

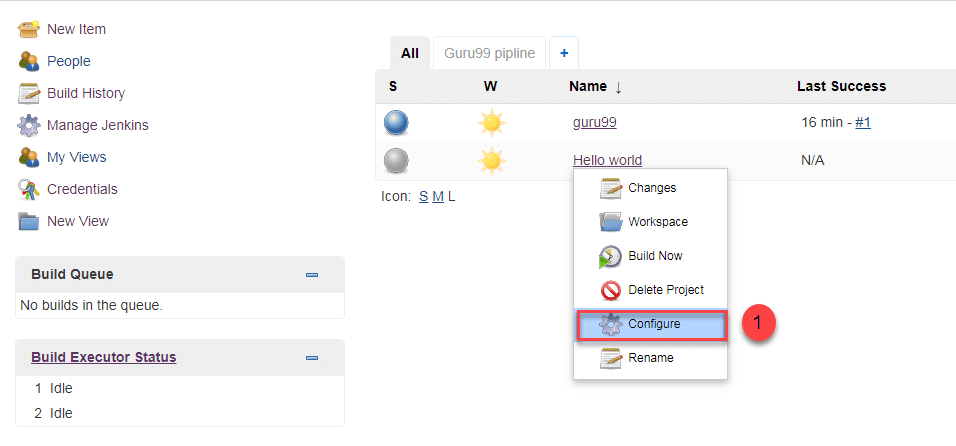
[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe6.png)

This will show you the sample pipeline view of your item, as given below:

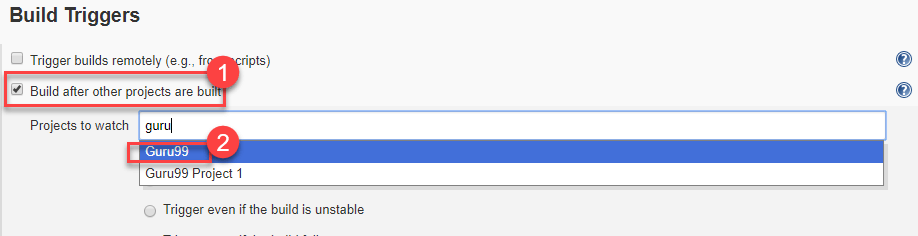
[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe7.png)

## Running a pipeline build

**Step 1**) For running a pipeline build, you need to chain your jobs first. For this, go to your first job and click on configure.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe8.png)

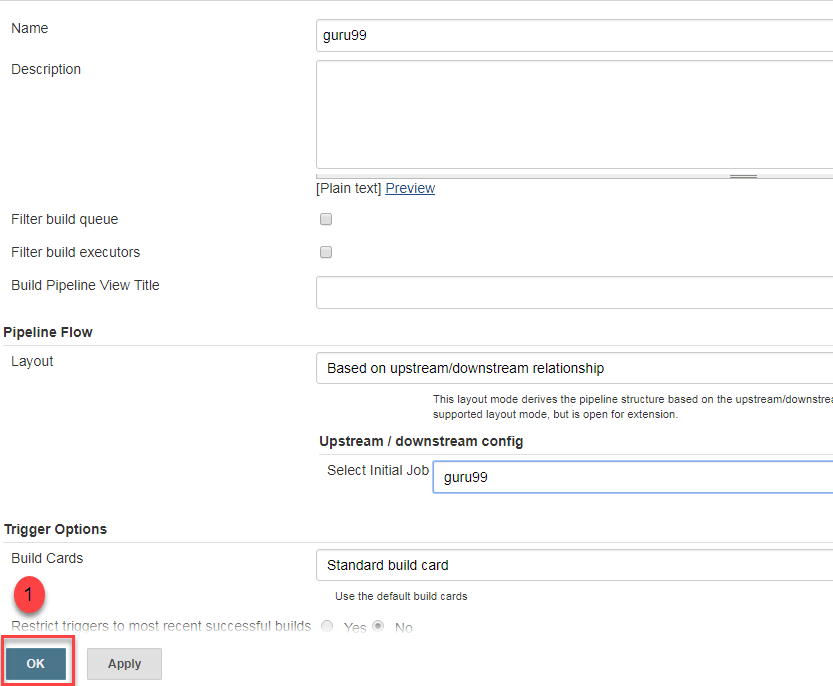
**Step 2**) Now, under **Build Triggers**, check the **Build after other projects are built** option.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe9.png)

Thus, a chain for all your jobs has been created.

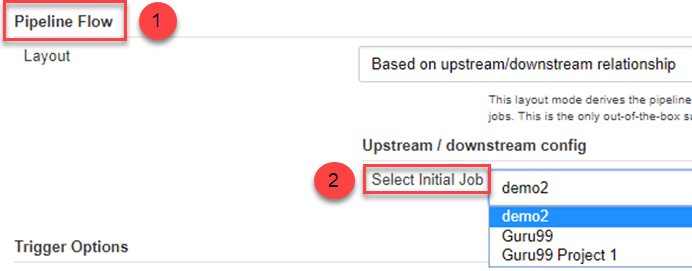
**Step 3**) Install the **Build Pipeline view** plugin if you don't have it installed already.

**Step 4**) Go to your Jenkins dashboard and create a view by clicking on the "**+**" button. Select the **Build Pipeline View** option and click **OK**.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe10.png)

**Step 5**) Under **Pipeline view configuration**, locate **Pipeline Flow**.

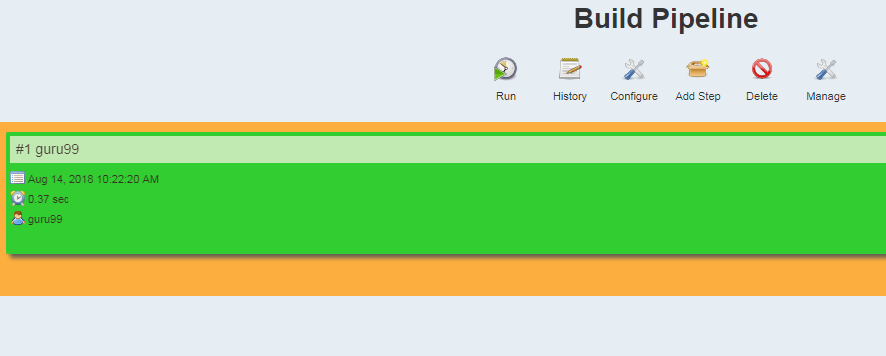
Under **Pipeline flow**, select the initial job to run. Now choose the job which has chains to other jobs, as shown in **Step 1** and **Step 2**.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe11.png)

Here we have selected **Guru99 Project 1** as the initial job, chained to other jobs. So, one by one, the jobs will run in the pipeline.

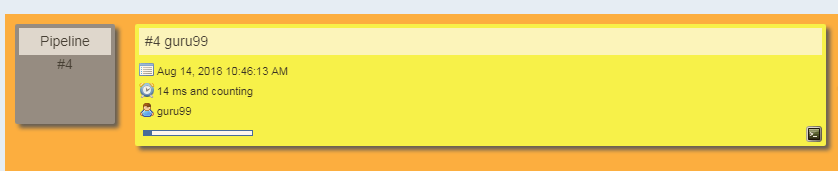
When the Jenkins pipeline is running, you can check its status with the help of Red and Green status symbols. Red means the pipeline has failed, while green indicates success.

In this example, we see that the button is green. Hence, the pipeline is successful.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe12.png)

## Running Jenkins pipeline

Click on **Run**to run the Jenkins pipeline. It will look something like this:

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe13.png)

In the example above, we are demonstrating a simple "helloworld.java" program. But in real time projects, you will be responsible for creating and building complex pipelines in Jenkins. See below for a sample pipeline view.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe14.png)

**Best Practices using Jenkins Pipeline:**

* Use the genuine Jenkins Pipeline
* Develop your pipeline as code
* Any non-setup work in your pipeline should occur within a stage block.
* Any material work in a pipeline must be performed within a node block.
* Don't use input within a node block.
* Never set environment variables with env global variable
* Wrap your inputs in a timeout

# Continuous Integration - Jenkins vs Travis-CI

## What is CI?

Continuous integration is a software development method where members of the team can integrate their work at least once in a day. In this method, every integration is checked by an automated build to search the error. The CI concept was first introduced over two decades ago to avoid "integration hell," which happens when integration is put off till the end of a project.

In this tutorial, you will learn

* [What is CI?](https://www.guru99.com/jenkins-vs-travis.html#1)
* [What is Travis CI?](https://www.guru99.com/jenkins-vs-travis.html#2)
* [What is Jenkins?](https://www.guru99.com/jenkins-vs-travis.html#3)
* [What does Travis do?](https://www.guru99.com/jenkins-vs-travis.html#4)
* [What did Jenkin do?](https://www.guru99.com/jenkins-vs-travis.html#5)
* [Travis CI Features:](https://www.guru99.com/jenkins-vs-travis.html#6)
* [Jenkin Features:](https://www.guru99.com/jenkins-vs-travis.html#7)
* [Travis vs. Jenkins](https://www.guru99.com/jenkins-vs-travis.html#8)
* [Popularity Index](https://www.guru99.com/jenkins-vs-travis.html#9)
* [Which is better?](https://www.guru99.com/jenkins-vs-travis.html#10)

**How CI works?**

* Developers write code and commit changes to the shared repository
* After that, the CI server monitors the repository and evaluates all the changes
* CI builds the system and conduct integration and unit tests
* The server releases deployable artifacts
* The Continous integration server assigns a build tag to the version and building code
* Then the CI server reports the team about the successful build. If the tests fail, the server alerts about the event to the development team. The team will fix the issues as fast as is possible.

## What is Travis CI?

Travis CI was the first CI as a Service tool. It introduced a new approach to building code in the cloud. This CI tool allows the user to sign up, link their repository, build, as well as test their apps.

Travis CI tool can easily integrate with the common cloud repositories like GitHub and Bitbucket. It offers many automated CI options which cut out the need for a dedicated server as the Travis CI server is hosted in the cloud. This allows you to test in different environments, on various machines, running on different Operating Systems.

[Travis CI](https://travis-ci.org/) is free for open source projects. For commercial projects, you need to purchase an enterprise plan.

## What is Jenkins?

Jenkins is an award-winning continuous integration tool that monitors executions of deployment cycles. It started as a side project by Sun's software engineers group. Later it was expanded as one of the popular open source CI tools which help software development teams to automate their deployments.

Jenkins is a Java-based tool, which means you only need Java Runtime Environment to operate it. Hence, Jenkins can be installed on any operating system where Java runs.

In this tool, Developers can also specify conditions for customized builds. Jenkins supports a massive plugin archive. This allows developers to alter how Jenkin looks and operates.

Moreover, the Jenkins Pipeline suite of plugins comes with special tools that allow developers to model easy-to-complex delivery pipelines using DSL ( Digital Subscribe line) method.

## What does Travis do?

Travis CI offers following benefits:

* You can monitor GitHub projects
* Runs Test and generate results quickly. Parallel test execution is possible.
* Build artifacts & check code quality
* Easy Deployment to cloud services
* It can identify small as well as large code changes.
* Developers can use Travis CI to watch the tests when they are running.
* The tool integrates with Slack, HipChat, Email, etc.

## What does Jenkin do?

Jenkins allows you to automate your build, test, and deploy tasks. The tool provides support for different OS like Windows, Mac OSX, and Linux systems.

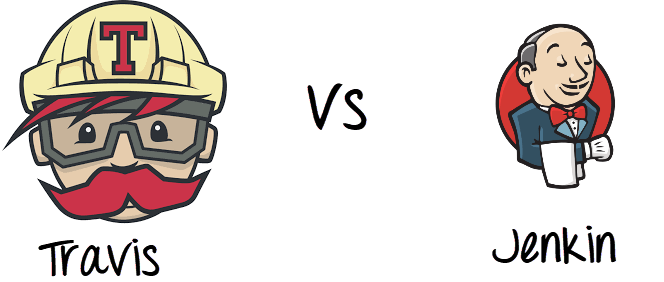
Moreover, Jenkins gives you an ability to quickly build and test your code to get early feedback on whether it's ready for production or not. In most cases, Jenkin will require few modifications according to your team's custom requirements.

## Travis CI Features:

* Automatic integration with GitHub
* Repository access to build pull requests
* Support for 21 languages like Android, C, C#, C++, Java, JavaScript (with Node.js), Perl, PHP, Python, R, Ruby, etc
* Pre-installed build & test tools
* Available services - databases, message queues, etc.
* Deployment to multiple cloud services
* Encrypt secure environment variables or files
* Virtual machines recreated after every build
* CLI client and API for scripting
* Comes with free cloud-based hosting which does not require maintenance or administration.

## Jenkin Features:

* Easy to install, upgrade, and configure
* Distributed Builds
* Monitoring external jobs
* More than 600 plugins to customize your Jenkins environment
* Over 1000+ public repositories on Github, 500+ contributors, strong commit activity
* Support for various authentication methods, version control systems, notification, etc.
* Jenkins provides remote access API and its functionalities.
* Provide Powerful CI/CD tool for big projects
* It supports various job models like Freestyle, Pipeline, etc.,
* Allows developers to add their extensions
* Compatible with Docker, Libvirt, Kubernetes, and many other programs

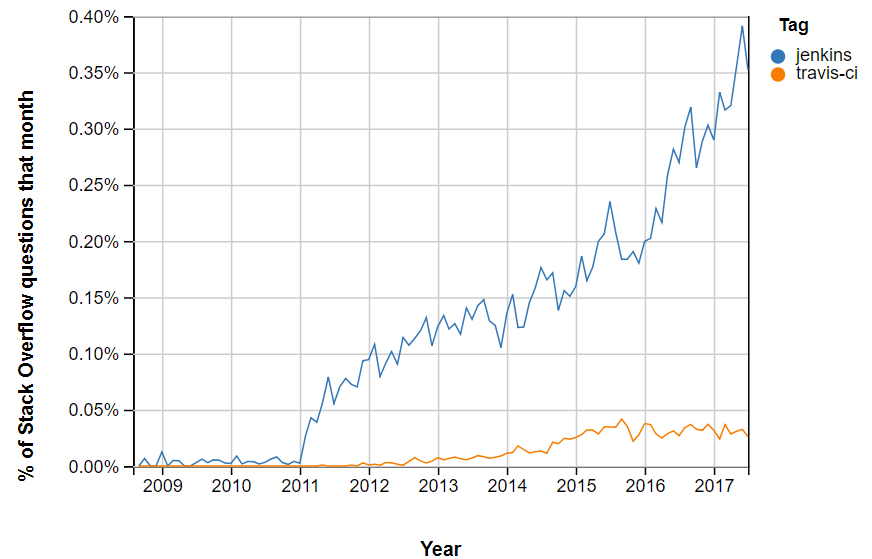
[](https://www.guru99.com/images/1/063018_1008_ContinuousI1.png)

## Travis vs. Jenkins

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Jenkin** | **Travis** |
| **Cost** | Jenkins is free. But development team need to run and maintain their dedicated server. This could be considered an extra expense. | Travis CI enterprise suites start at $129 per month. Cost increase based on the level of support you require. |
| **Set up Time** | Jenkins needs elaborate setup. So you'll have a very long wait time for the complete installation. | It takes very less time to get started. Create a config file and start integrating. |
| **Performance** | If you're looking for a CI tool with unlimited customization options, then Jenkins is the best choice for you. | Travis CI is the best choice If you are working in an open source project. |
| **Tool Type** | It is an open-source free to use the tool. | It is a commercial CI Tool |
| **Usage** | Easy to use | Flexible to use |
| **Github** | Good for Github | Excellent for Github |
| **Support** | Extensive support from the community. | Limited support for the community. |
| **Pros** | * Customization is biggest benefit of the Jenkins CI server * An extensive plugin archive can be configured to change the look of Jenkins. * You can also add new functionality like authentication, alerts, and credentials. | * Integration with GitHub & cloud * Unlimited open source projects with full functionality * Extensive project configuration via .travis.ymi file * Allows cluster tests and run them in parallel * Multiple build environments and target platforms (i.e. Node 0.10,0.8,0.6, Li on). |
| **Cons** | * One major drawback of Jenkins for continuous integration is that it is highly configurable. There is no out-of-the-box setup. That's why it may take two-three hours to days to get everything configured in the system. | * The biggest cons of installing Travis CI is that it's Commercial plans start at $129/m which is quite expensive. * Not suitable for high-security projects * Unlike other CI tools, it does not offer Bitbucket Support. |
| **Usage Plans** | Free | Free for open source projects. However, Paid for Enterprise. |
| **Server Machine** | Server-based | Cloud-based |
| **Customization Options** | More | Less |
| **Configuration** | Fully customizable | YAML |
| **Control on system** | Full | Very less |

## Popularity Index

The number of questions labeled Jenkins and Travis in Stack Overflow.

[](https://www.guru99.com/images/1/063018_1008_ContinuousI2.png)

## Which is better?

Thus, with the above discussion, we can get ay that Travis and Jenkins both offer wonderful features. However, small open source projects are best suited for Travis CI as it is easy to run and quick to set up. On the other hand, large enterprise is best suited to Jenkins as it offers free licensing for a private project and a wide range of customizable feature. So, we can say that both of these continuous integration tools are good in their way.