

Jenkins GitHub Integration: Install Git

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>.

In this you will learn:

- Installation of Plugins in Jenkins
- Install GIT Plugin
- Integrating Jenkins with GitHub

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.

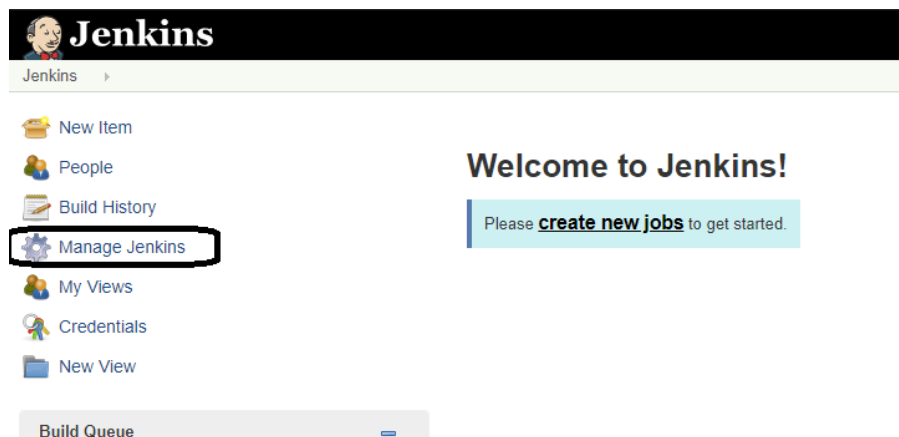
GitHub is a web-based repository of code which plays a major role in DevOps. It provides a common platform for multiple developers working on the same code/project to upload and retrieve updated code, thereby facilitating continuous integration.

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

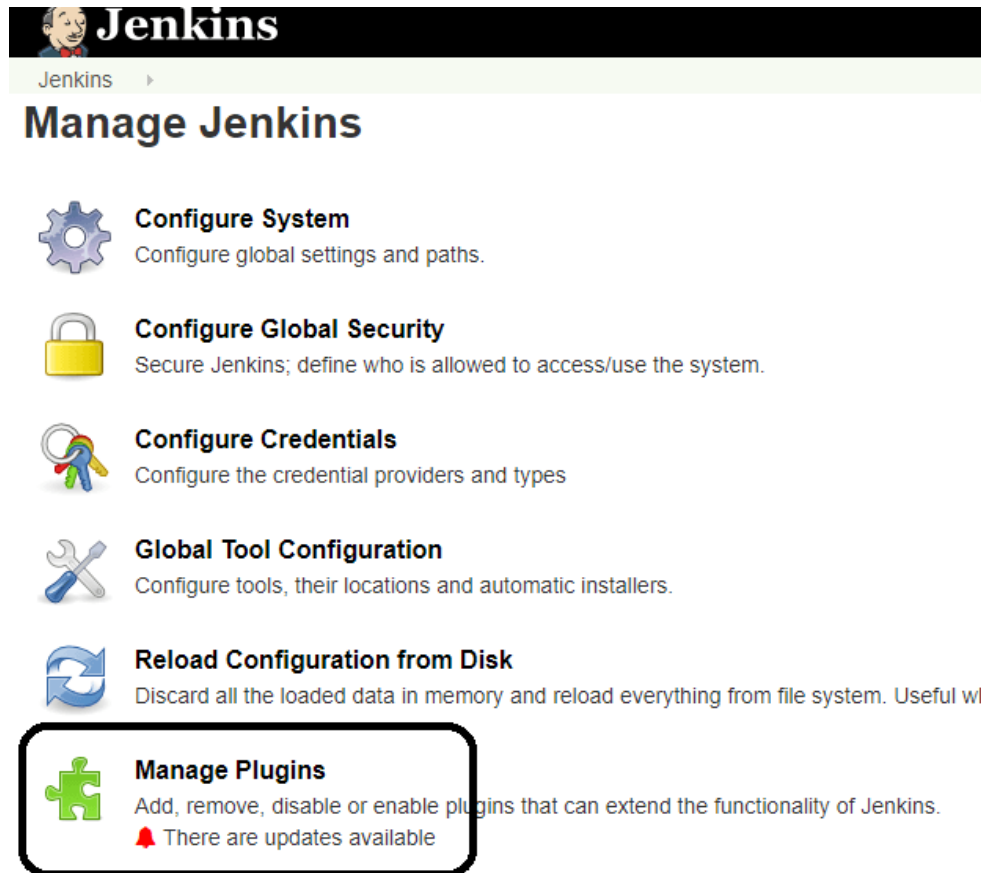
You need not install a GitHub plugin if you have already installed the Git plugin in response to the prompt during the Jenkins' installation setup. But if not, here is how you install GitHub plugins in Jenkins and pull code from a GitHub repository.

Install GIT Plugin

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



Step 2: Click on Manage Plugins:



github integrationStep 3: In the Plugins Page

1. Select the ***GIT Plugin, GitHub and GitHub Integration 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.

Team Concert Git Plugin
Integrates Jenkins with [Rational Team Concert](#) for Jenkins Builds which use Git as source control. This plugin will create traceability links from a Jenkins build to Rational Team Concert [Work Items](#) and [build](#) results. This plugin adds traceability links from a Jenkins build to an RTC build result. It also publishes links to work items and annotates the change log generated by Jenkins with links to RTC Work Items; It leverages the current RTC features and workflows that users are already familiar with such as, emails, toaster popups, reporting, dashboards, etc. 1.0.9

Tracking Git Plugin
Lets one project track the Git revisions that are built for another project. 1.0

Git Plugin
This plugin allows use of [Git](#) as a build SCM. A recent Git runtime is required (1.7.9 minimum, 1.8.x recommended). Plugin is only tested on official [git client](#). Use exotic installations at your own risks. 2.3.5

Repo Plugin
This plugin adds Repo (<http://code.google.com/p/git-repo/>) as an SCM provider in Jenkins. 1.6

Embeddable Build Status Plugin
This plugin allows Jenkins to expose the current status of your build as an image in a fixed URL. You can put this URL into other sites (such as GitHub README) so that people can see the current state of the job (last build) or for a specific build. 1.6

2 Install without restart **3** Download now and install after restart **4** Check now
Update information obtained: 10 min ago

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.

☒ **GitHub plugin** 1.29.1 Uninstall
This plugin integrates [GitHub](#) to Jenkins.

☒ **GitHub Branch Source Plugin** 2.3.6 Uninstall
Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.

☒ **GitHub API Plugin** 1.92 Uninstall
This plugin provides [GitHub API](#) for other plugins.

☒ **GIT server Plugin** 1.7 Uninstall
Allows Jenkins to act as a Git server.

☒ **Git plugin** 3.9.1 Uninstall
This plugin integrates [Git](#) with Jenkins.

☒ **Git client plugin** 2.7.2 Uninstall
Utility plugin for Git support in Jenkins

Integrating Jenkins with GitHub

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

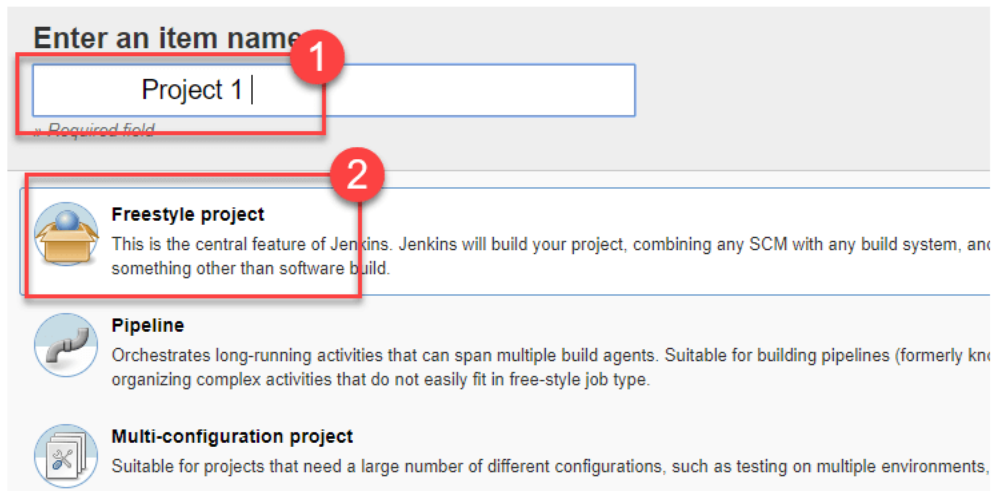
Step 1) Create a new job in Jenkins, open the Jenkins dashboard with your Jenkins URL. For example, <http://localhost:8080/>

Click on **create new jobs**:

Welcome to Jenkins!

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

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



Enter an item name

Project 1 |

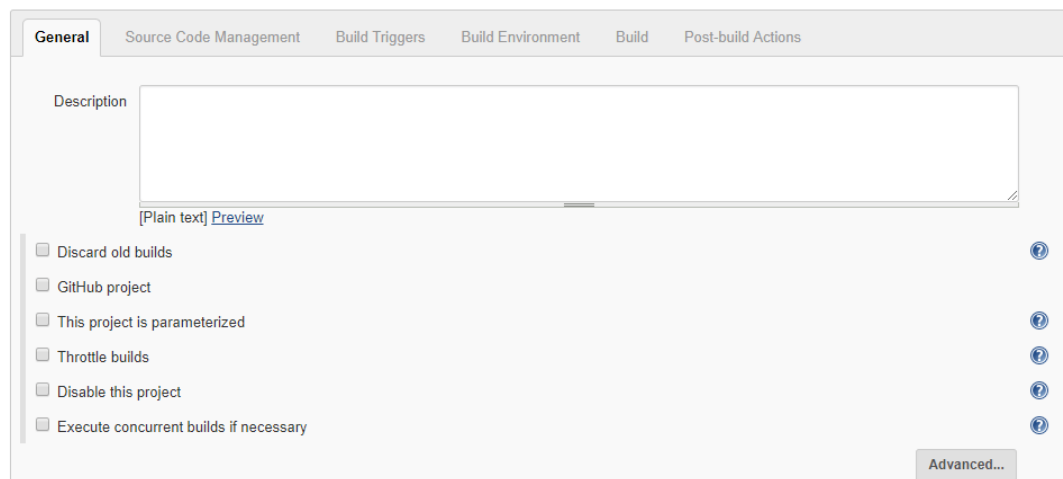
Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as declarative pipeline) 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, or building different versions of a project.

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



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

Description

[Plain text] [Preview](#)

☐ Discard old builds

☐ GitHub project

☐ This project is parameterized

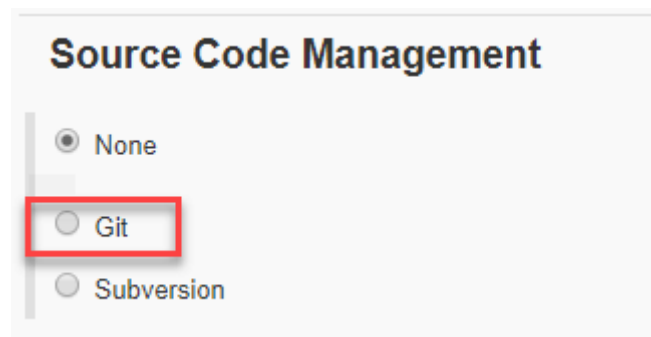
☐ Throttle builds

☐ Disable this project

☐ Execute concurrent builds if necessary

[Advanced...](#)

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



Source Code Management

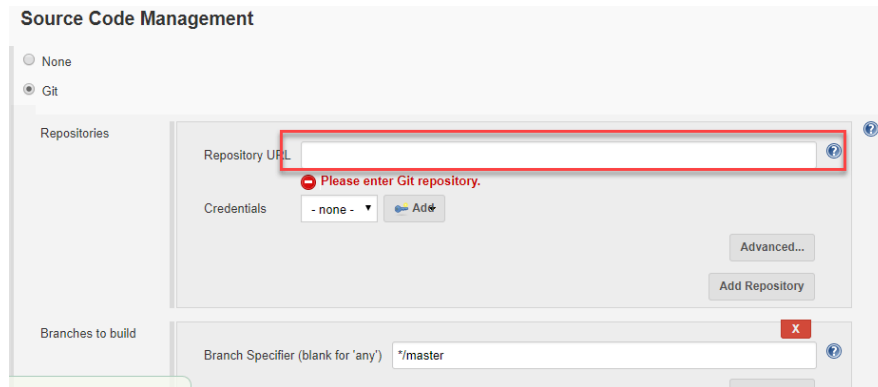
☒ None

☐ Git

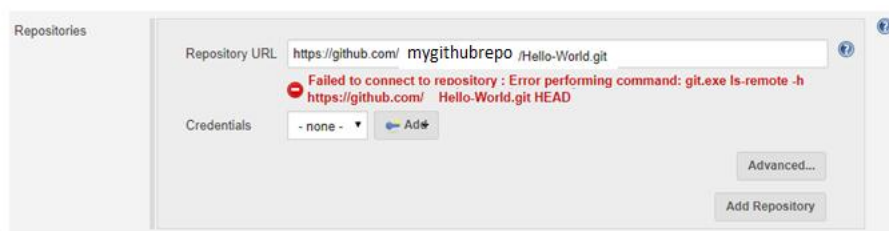
☐ Subversion

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.



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



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



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.

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


```
Microsoft Windows [Version 10.0.16299.309]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\alex>git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]

These are common Git commands used in various situations:
```

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:



The screenshot shows the Jenkins 'Source Code Management' configuration interface. On the left, there are two radio buttons: 'None' and 'Git', with 'Git' being the selected option. Below this is a section titled 'Repositories'. Inside this section, there is a 'Repository URL' field containing the text 'https://github.com/leereilly/hello-world-java.git' and a 'Credentials' dropdown menu currently showing '- none -'. To the right of the dropdown is a small 'Add' button with a key icon. At the bottom right of the 'Repositories' section, there are two buttons: 'Advanced...' and 'Add Repository'.

Git is now properly configured on your system.