



Jenkins

Running Freestyle Project

JENKINS

BY ALOK SRIVASTAVA - NETWORK NUTS



FREESTYLE PROJECT

Freestyle means improvised or unrestricted. A freestyle project in Jenkins is a project that spans multiple operations. It can be a build, a script run, or even a pipeline. According to the official Jenkins wiki, a freestyle project is a typical build job or task.



This could be as simple as running tests, building or packaging an application, sending a report, or even running some commands. Before any tests are run, data is collated. This can also be done by Jenkins. Jenkins collects data through multiple ways depending on what is being achieved and the purpose of the data in question. A real-world scenario could involve, for instance, the collection of application artifacts after builds.

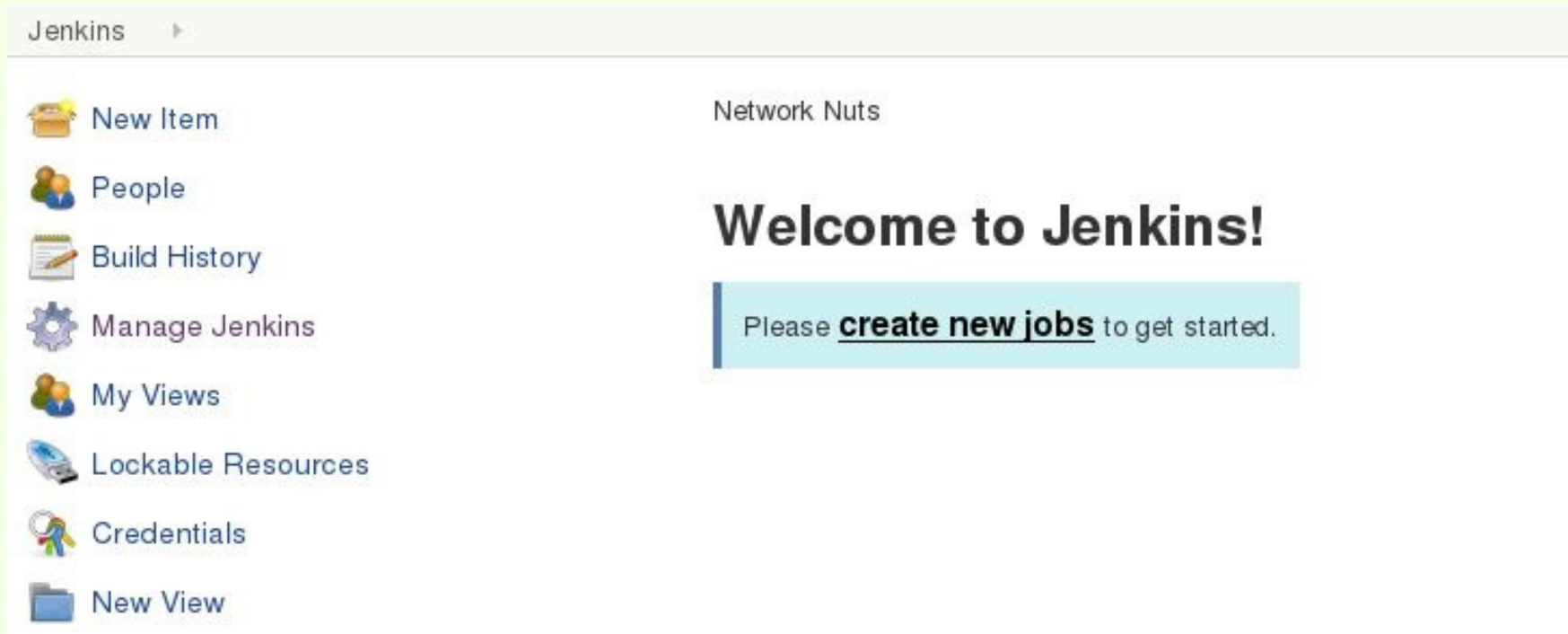
In relation to management, Jenkins allows us to send reports at any defined stage, which could entail artifact information, or shipping application logs to a log management entity, such as Elasticsearch.

Running Freestyle Project

We have a sample bash script available inside our git account. Copy the name and the script name "hello.sh", in our case.

The screenshot shows the GitHub interface for the repository 'alokaryan / checkjenkins'. At the top, there are buttons for 'Unwatch', 'Star' (0), and 'Fork' (0). Below this is a navigation bar with 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. The repository name 'testing integration with jenkins' is displayed with an 'Edit' button. Below the name, there's a 'Manage topics' link. A summary bar shows '3 commits', '1 branch', '0 releases', '1 contributor', and 'Apache-2.0' license. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. A dropdown menu for 'Clone or download' is open, showing 'Clone with HTTPS' (selected) and 'Use SSH'. The HTTPS URL 'https://github.com/alokaryan/checkjenkins' is highlighted. Below the URL are buttons for 'Open in Desktop' and 'Download ZIP'. On the left, a file list shows 'LICENSE' (Initial commit), 'README.md' (Initial commit), and 'hello.sh' (more changes done).

Open the main dashboard to create a project using the "New Item" option on the left navigation menu.




Give your project some name, select "freestyle project" and select "OK".

Enter an item name


testgit

» 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.



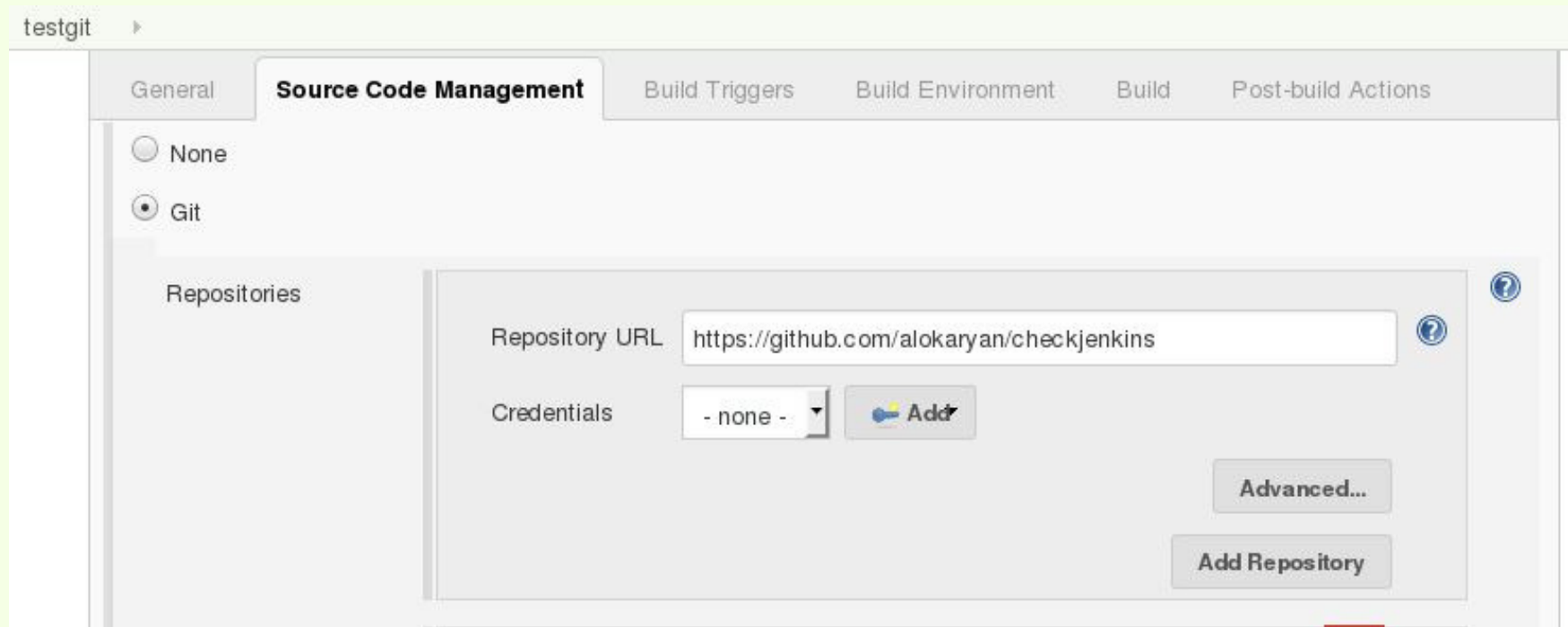
Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

OK

Folder

In the "Source Code Management" tab paste the url of the git repository, copied earlier.



The screenshot shows the Jenkins configuration interface for a job named "testgit". The "Source Code Management" tab is selected, displaying options for repository management. The "Git" option is chosen. Under the "Repositories" section, the "Repository URL" is set to "https://github.com/alokaryan/checkjenkins". The "Credentials" dropdown is set to "- none -", with an "Add" button next to it. At the bottom right of the configuration area, there are buttons for "Advanced..." and "Add Repository".

testgit ▶

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

☐ None
☒ Git

Repositories

Repository URL

Credentials

In the "Build" section select "execute shell" and type the command to execute our bash shell script.



Build

Execute shell

Command `sh hello.sh`

[See the list of available environment variables](#)

Advanced...

After your select "apply and save" you can see your project appearing on dashboard.

Jenkins [ENABLE AUTO REFRESH](#)

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

Network Nuts [edit description](#)

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		testgit	N/A	N/A	N/A 

Icon: [S](#) [M](#) [L](#)

[Legend](#)  [RSS for all](#)  [RSS for failures](#)  [RSS for just latest builds](#)

Click on the "down arrow" besides the name "testgit" and select "build now". If everything is perfect, you can see the blue icon.

Jenkins [DISABLE AUTO REFRESH](#)

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

Network Nuts [edit description](#)

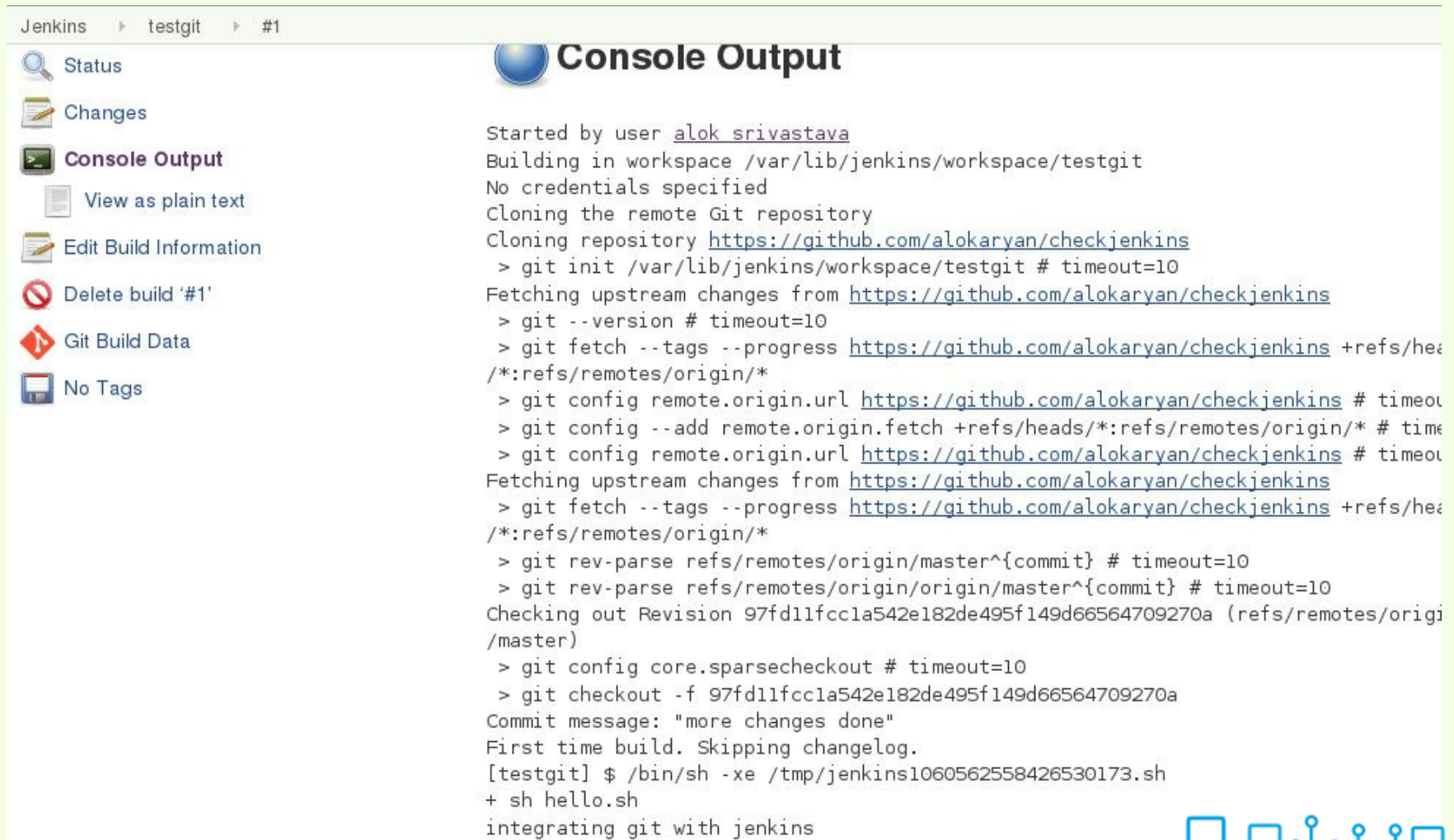
All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		testgit	28 sec - #1	N/A	4.3 sec 

Icon: [S](#) [M](#) [L](#)

[Legend](#)  [RSS for all](#)  [RSS for failures](#)  [RSS for just latest builds](#)

You can also check what exactly happened by select the "build number" (1, in our case) and selecting "console outout"



The screenshot shows the Jenkins web interface for a build named 'testgit' with build number '#1'. The left sidebar contains links to 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', 'Delete build '#1'', 'Git Build Data', and 'No Tags'. The main area is titled 'Console Output' and displays the following text:

```
Started by user alok srivastava
Building in workspace /var/lib/jenkins/workspace/testgit
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/alokaryan/checkjenkins
> git init /var/lib/jenkins/workspace/testgit # timeout=10
Fetching upstream changes from https://github.com/alokaryan/checkjenkins
> git --version # timeout=10
> git fetch --tags --progress https://github.com/alokaryan/checkjenkins +refs/heads/*:refs/remotes/origin/*
> git config remote.origin.url https://github.com/alokaryan/checkjenkins # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/alokaryan/checkjenkins # timeout=10
Fetching upstream changes from https://github.com/alokaryan/checkjenkins
> git fetch --tags --progress https://github.com/alokaryan/checkjenkins +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 97fd11fcc1a542e182de495f149d66564709270a (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 97fd11fcc1a542e182de495f149d66564709270a
Commit message: "more changes done"
First time build. Skipping changelog.
[testgit] $ /bin/sh -xe /tmp/jenkins1060562558426530173.sh
+ sh hello.sh
integrating git with jenkins
```


Now, let's get back to examining what the rest of the tabs in the project configuration view do. The **Build Triggers** resource helps in automating builds. When setting up pipelines, some of the processes need to be automated in order to be effective.

When changes are pushed to GitHub, Jenkins should automatically run tests and build applications instead of developers triggering manual builds each time.

Build Triggers

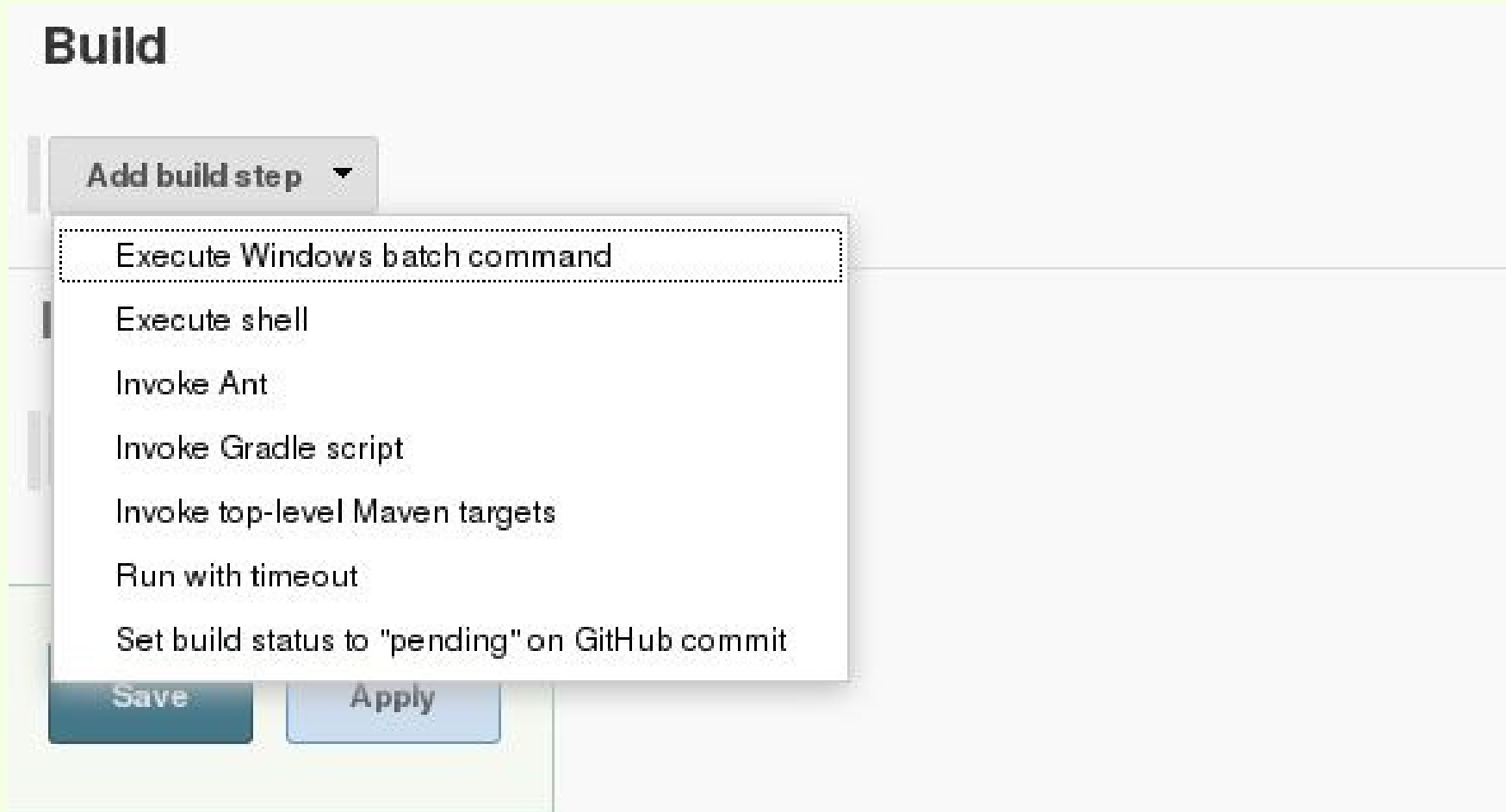
- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

Build Environment resource, is involved with the environment—more precisely, the application environment. Credentials need to be set to, for example, access a server; a language-specific detail, such as Python virtual environments; and project management resources such as ticket etc.

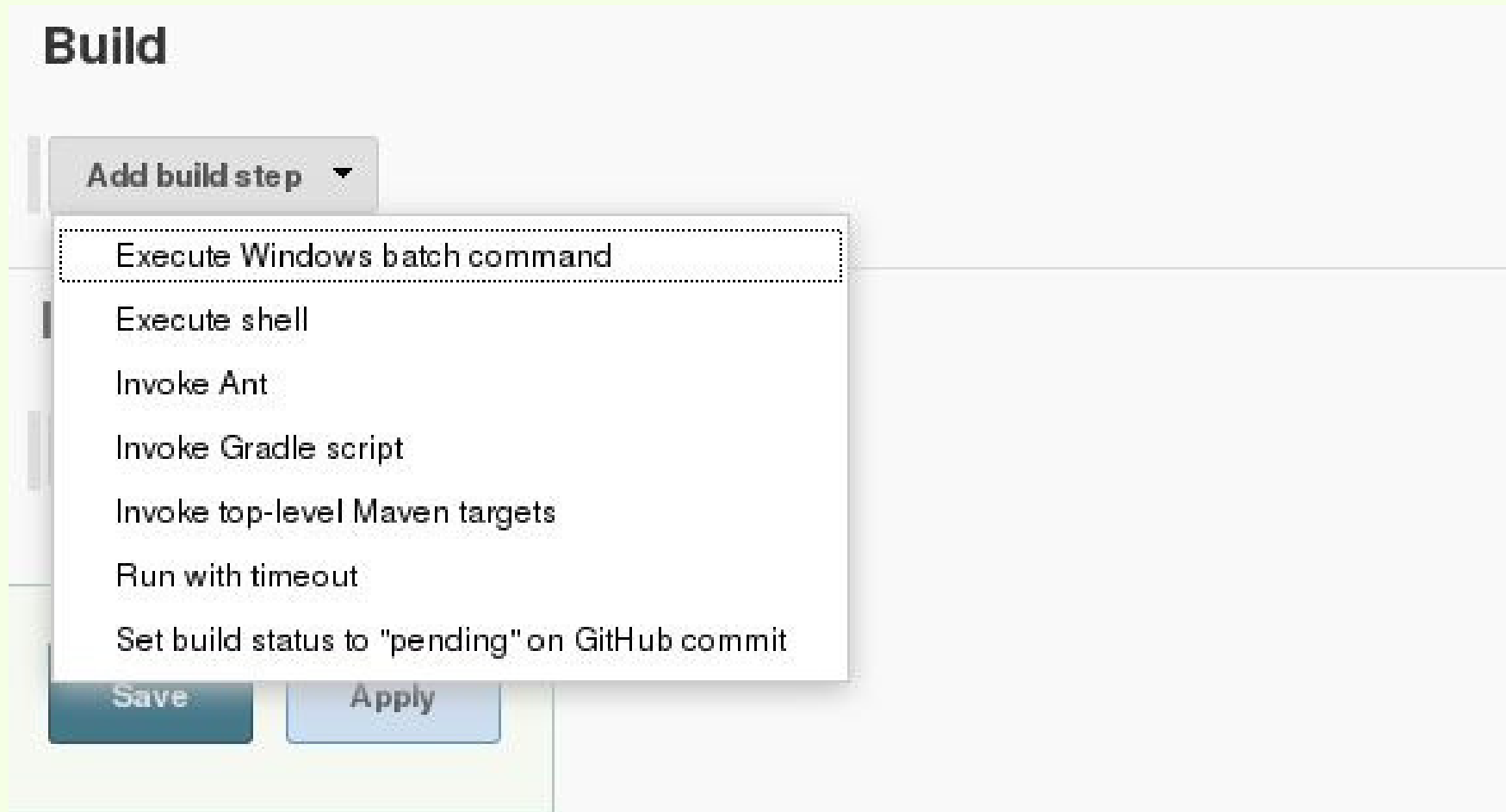
Build Environment

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published Gradle build scans
- ☐ With Ant

Build resource defines the actual steps we want to achieve



Build resource defines the actual steps we want to achieve



We can also check the complete build history of our jobs.

Jenkins ▸ All ▸ [DISABLE AUTO REFRESH](#)

[New Item](#)

- [People](#)
- [Build History](#)**
- [Manage Jenkins](#)
- [My Views](#)
- [Lockable Resources](#)
- [Credentials](#)
- [New View](#)

Build Queue ☐

No builds in the queue.

Build Executor Status ☐

1 Idle

2 Idle

Build History of Jenkins



[Export as plain XML](#) ([Recurse in subfolders](#))

Build	Time Since ↑	Status
 testgit #2	5 min 58 sec	stable 
 testgit #1	45 min	stable 

In case you are trying to run a python build. You need to install python plugin first.

Filter:

Updates Available Installed Advanced

Install	Name	Version
<input checked="" type="checkbox"/>	Python Adds the ability to execute python scripts as build steps.	1.3
<input type="checkbox"/>	ShiningPanda This plugin adds Python support to Jenkins with some useful builders (standard Python builder, virtualenv builder, ...) and the ability to use a Python axis in multi-configuration projects (for testing on multiple versions of Python).	0.24
<input type="checkbox"/>	Python Wrapper This plugin provides the runtime library for plugins written in Python.	1.0.3
<input type="checkbox"/>	Pyenv Pipeline Allows a Durable Task (sh or bat) to be executed within a Python virtualenv	2.1.1-STAGING
Python Development		
<input type="checkbox"/>	pyenv Run Jenkins builds in pyenv	0.0.7

Install without restart

Download now and install after restart