



**[START TESTING NOW \(HTTPS://A.BLAZEMETER.COM/APP/SIGN-UP\)](https://a.blazemeter.com/app/sign-up)**



January 12, 2021

# How to Integrate Your GitHub Repository to Your Jenkins Project

OPEN SOURCE AUTOMATION

By Guy Salton

GitHub and Jenkins (<https://www.blazemeter.com/blog/jenkins-automation>) — can you use them together? And can you integrate them? Find out in this blog!

## Table of Contents:

- [Can You Use Jenkins With GitHub?](#)
- [Why Integrate GitHub + Jenkins?](#)
- [How to Integrate GitHub + Jenkins](#)

# Can You Use Jenkins With GitHub?

You can and should use Jenkins with GitHub to save time and keep your project up-to-date.

One of the basic steps of implementing CI/CD is integrating your SCM (Source Control Management) (<https://www.blazemeter.com/blog/scm-version-control>) tool with your CI tool. This saves you time and keeps your project updated all the time. One of the most popular and valuable SCM tools is GitHub. In this blog I will explain how to integrate Jenkins with GitHub projects.

## Why Integrate GitHub + Jenkins?

You should integrate GitHub and Jenkins to improve efficiency.

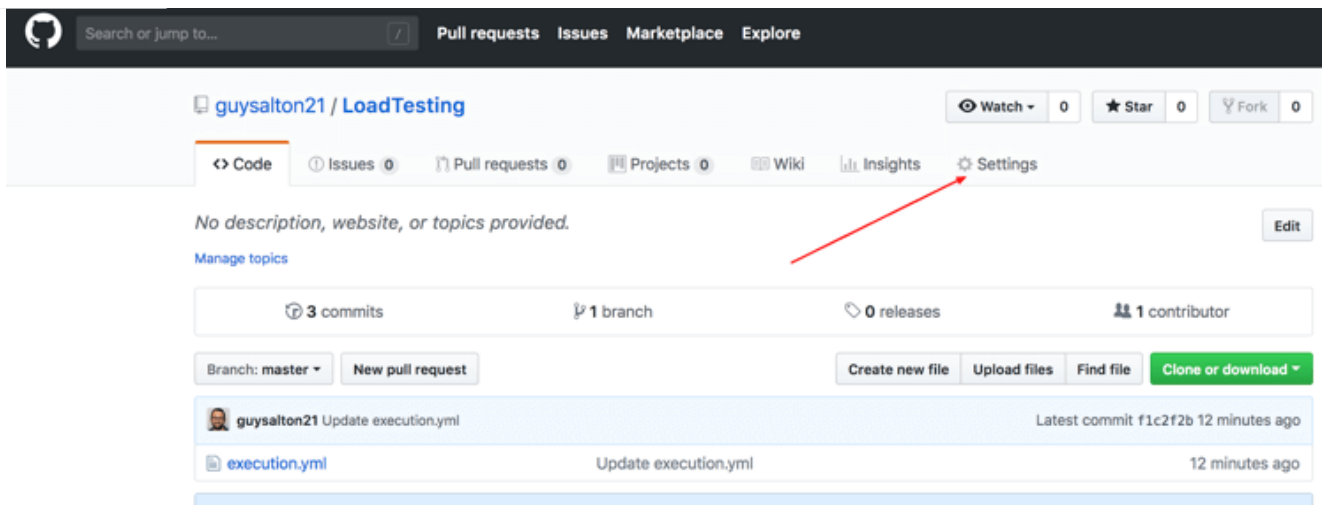
The integration presented in this blog post will teach you how to schedule your build, Pull your code and data files from your GitHub repository to your Jenkins machine, and automatically trigger each build on the Jenkins server after each Commit on your Git repository

But first, let's configure the GitHub and Jenkins integration. Let's begin with the GitHub side!

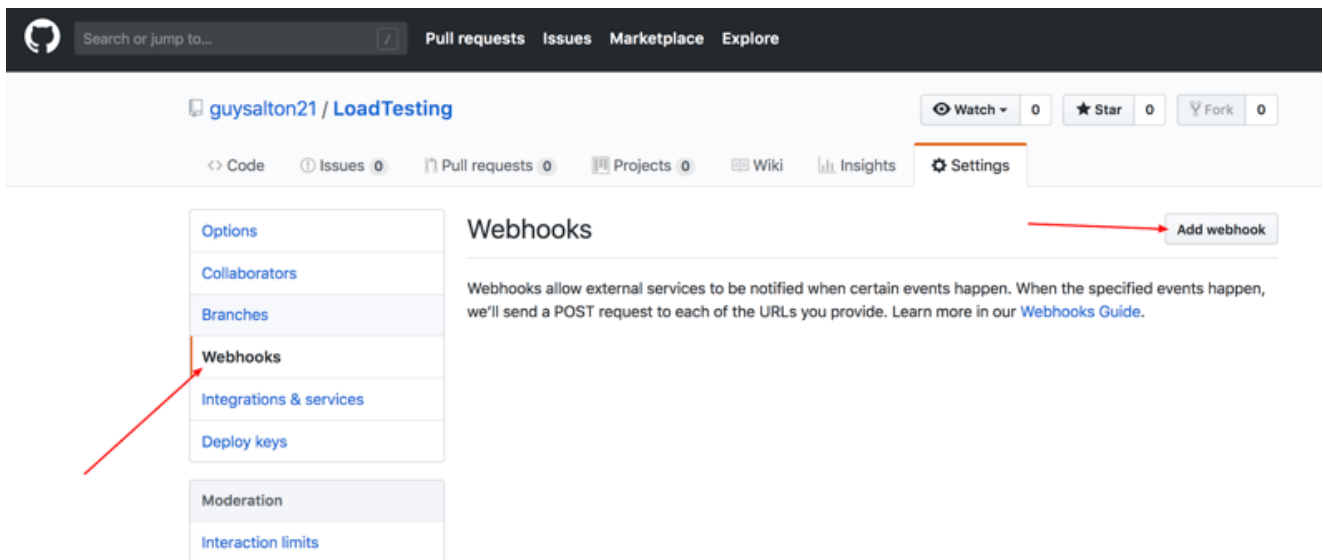
## How to Integrate GitHub + Jenkins

### Configuring GitHub

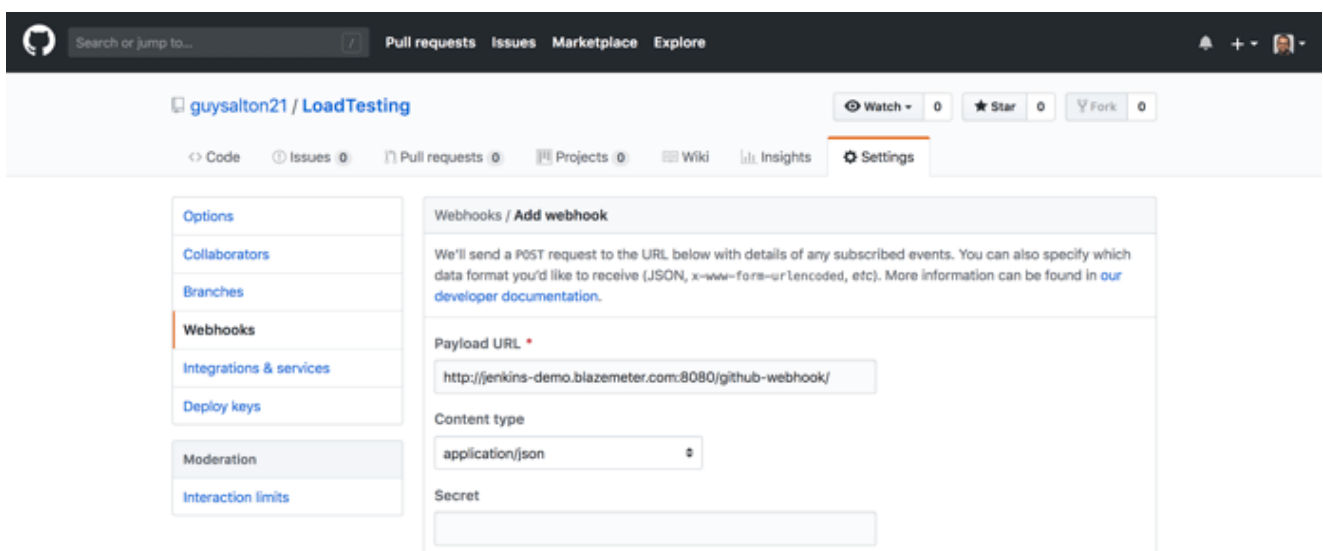
**Step 1:** go to your GitHub repository and click on '**Settings**'.



**Step 2:** Click on **Webhooks** and then click on **'Add webhook'**.



**Step 3:** In the **'Payload URL'** field, paste your Jenkins environment URL. At the end of this URL add `/github-webhook/`. In the **'Content type'** select: `'application/json'` and leave the **'Secret'** field empty.



#### Step 4: In the page 'Which events would you like to trigger this webhook?'

choose 'Let me select individual events.' Then, check 'Pull Requests' and 'Pushes'. At the end of this option, make sure that the 'Active' option is checked and click on 'Add webhook'.

Which events would you like to trigger this webhook?

☐ Just the push event.

☐ Send me everything.

☒ Let me select individual events.

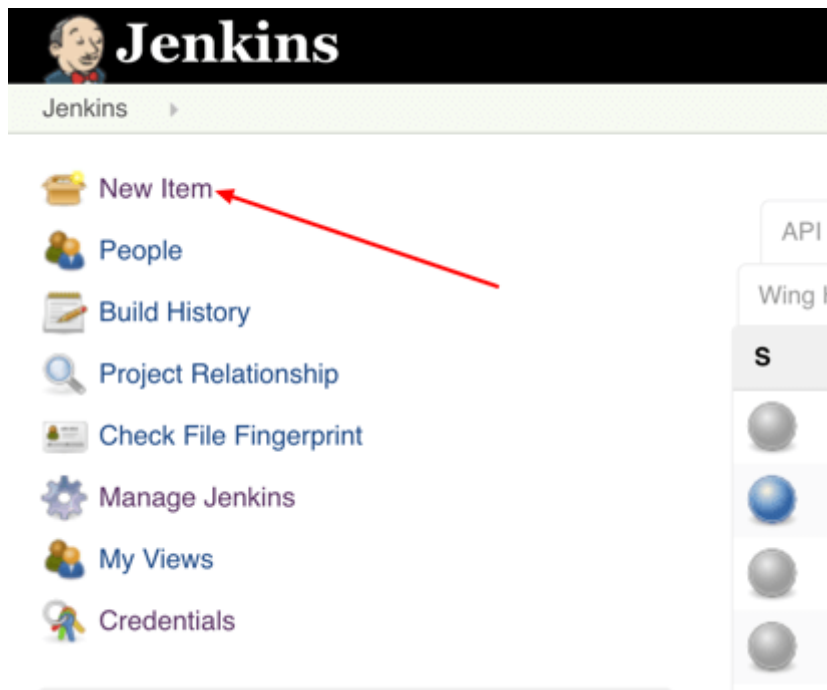
<input type="checkbox"/> Check runs Check run is created, requested, rerequested, or completed.	<input type="checkbox"/> Check suites Check suite is requested, rerequested, or completed.
<input type="checkbox"/> Commit comments Commit or diff commented on.	<input type="checkbox"/> Branch or tag creation Branch or tag created.
<input type="checkbox"/> Branch or tag deletion Branch or tag deleted.	<input type="checkbox"/> Deployments Repository deployed.
<input type="checkbox"/> Deployment statuses Deployment status updated from the API.	<input type="checkbox"/> Forks Repository forked.
<input type="checkbox"/> Wiki Wiki page updated.	<input type="checkbox"/> Issue comments Issue comment created, edited, or deleted.
<input type="checkbox"/> Issues Issue opened, edited, deleted, transferred, closed, reopened, assigned, unassigned, labeled, unlabeled, milestone, or demilestoned.	<input type="checkbox"/> Labels Label created, edited or deleted.
<input type="checkbox"/> Collaborator add, remove, or changed Collaborator added to, removed from, or has changed permissions for a repository.	<input type="checkbox"/> Milestones Milestone created, closed, opened, edited, or deleted.
<input type="checkbox"/> Visibility changes Repository changes from private to public.	<input checked="" type="checkbox"/> Pull requests Pull request opened, closed, reopened, edited, assigned, unassigned, review requested, review request removed, labeled, unlabeled, or synchronized.
<input type="checkbox"/> Pull request reviews Pull request review submitted, edited, or dismissed.	<input type="checkbox"/> Pull request review comments Pull request diff comment created, edited, or deleted.
<input checked="" type="checkbox"/> Pushes Git push to a repository.	<input type="checkbox"/> Releases Release published in a repository.
<input type="checkbox"/> Repositories Repository created, deleted, archived, unarchived, publicized, or privatized.	<input type="checkbox"/> Repository imports Repository import succeeded, failed, or cancelled.
<input type="checkbox"/> Repository vulnerability alerts Vulnerability alert created, resolved, or dismissed on a repository.	<input type="checkbox"/> Statuses Commit status updated from the API.
<input type="checkbox"/> Team adds Team added or modified on a repository.	<input type="checkbox"/> Watches User stars a repository.

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

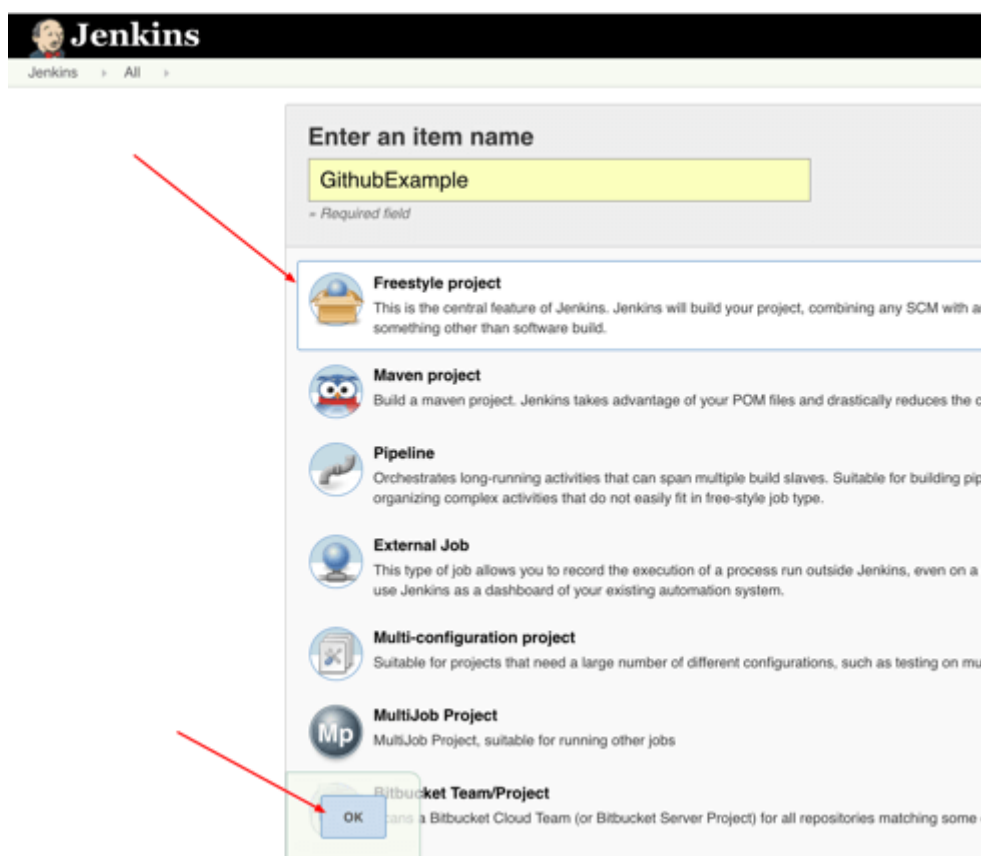
**Add webhook**

We're done with the configuration on GitHub's side! Now let's move on to Jenkins.

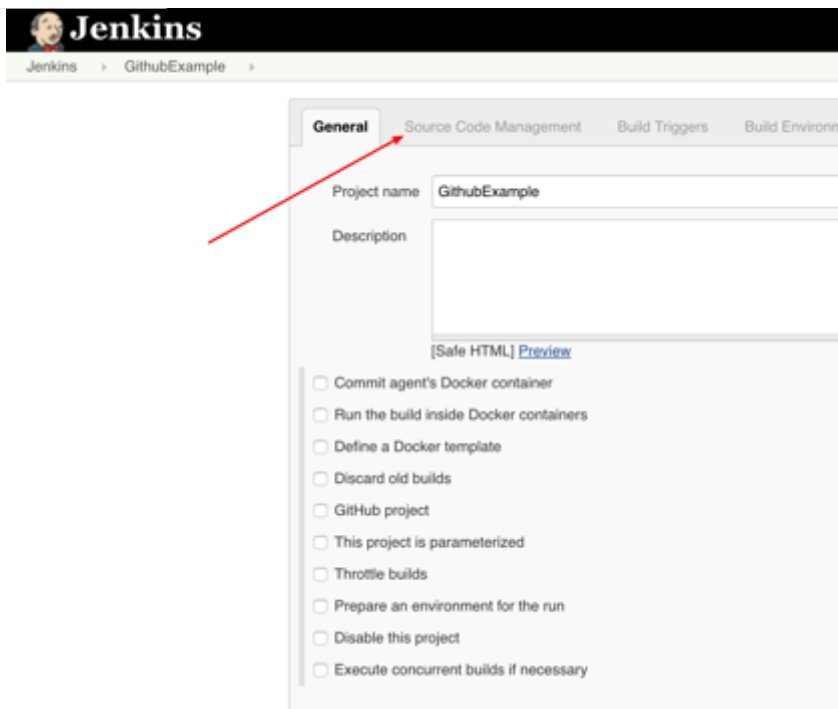
**Step 5:** In Jenkins, click on **'New Item'** to create a new project.



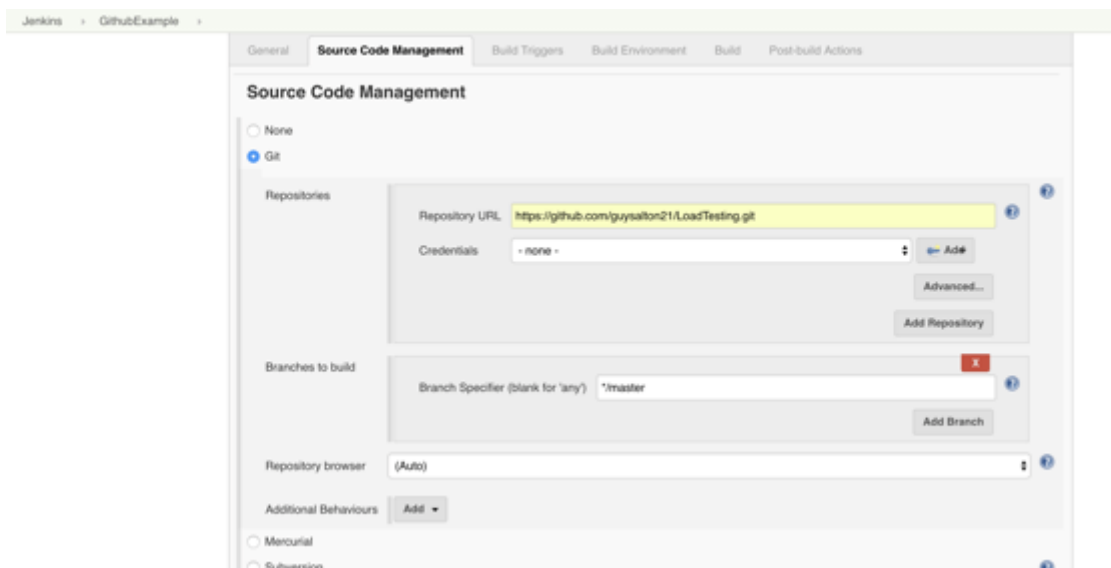
**Step 6:** Give your project a name, then choose **'Freestyle project'** and finally, click on **'OK'**.



**Step 7:** Click on the **'Source Code Management'** tab.

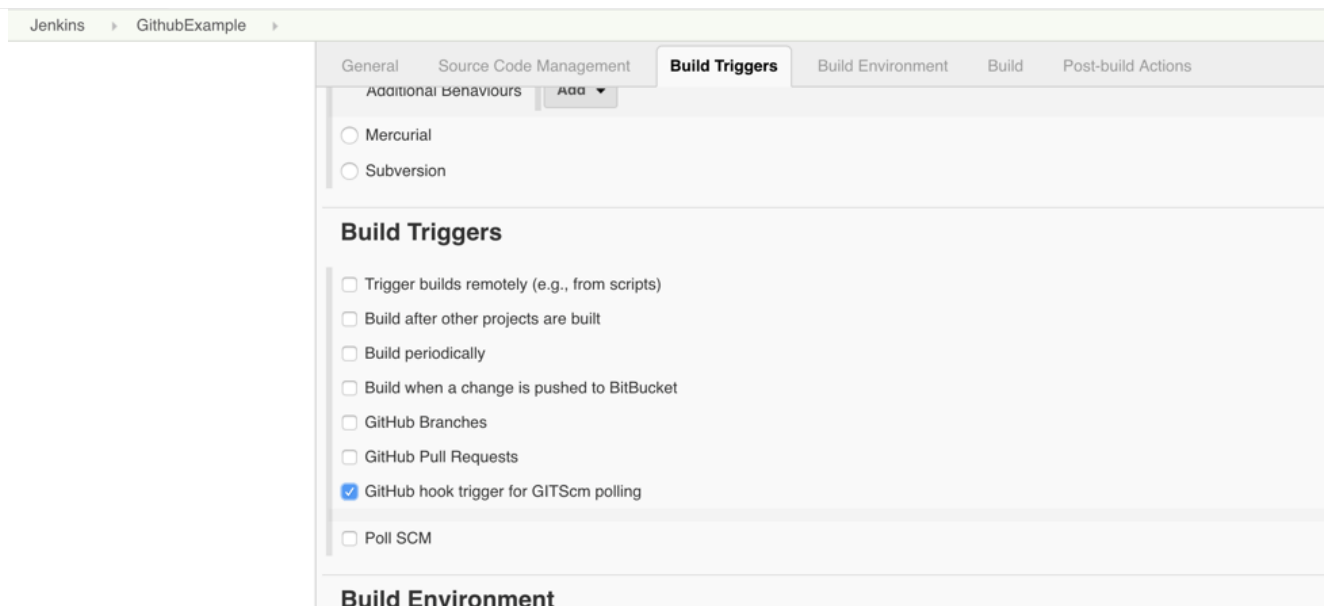


**Step 8:** Click on Git and paste your GitHub repository URL in the **'Repository URL'** field.



**Step 9:** Click on the **'Build Triggers'** tab and then on the **'GitHub hook trigger for GITScm polling'**. Or, choose the trigger of your choice.

...n/sh.  
...  
...og/i

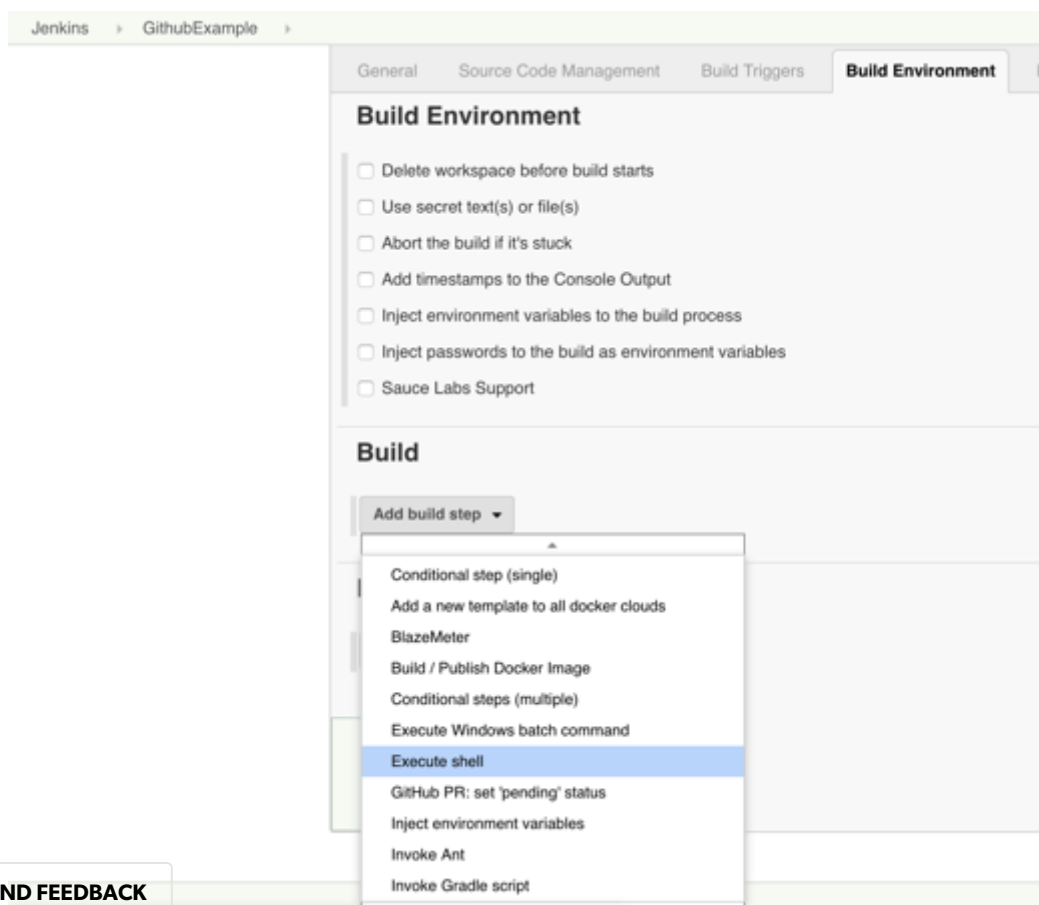


That's it! Your GitHub repository is integrated with your Jenkins project. You can now use any of the files found in the GitHub repository and trigger the Jenkins job to run with every code commit.

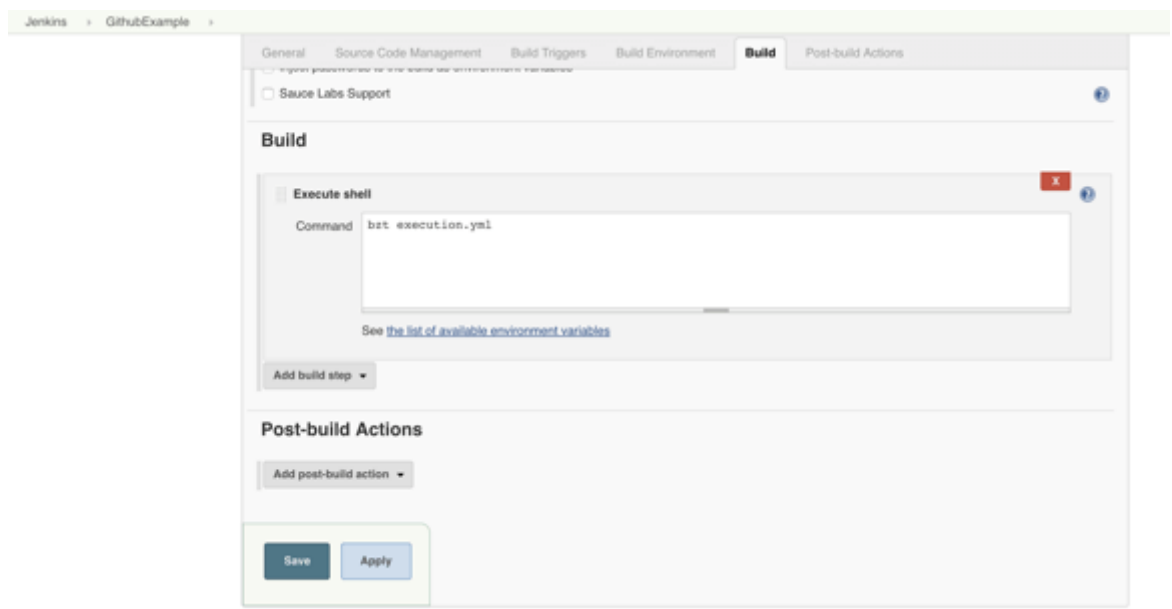
For example, I will show you how to run a Taurus script that I uploaded to my GitHub repository from my Jenkins project.

## Triggering the Jenkins GitHub Integration With Every Code Commit

**Step 10:** Click on the **'Build'** tab, then click on *'Add build step'* and choose *'Execute shell'*.



**Step 11:** To run a Taurus test, simply use the **'bzt'** command, followed by the name of your YML file and click on **'Save'**.



**Step 12:** Go back to your GitHub repository, edit the Taurus script and commit the changes. We will now see how Jenkins ran the script after the commit.



&lt;&gt; Edit file Preview changes

Spaces

```
1 ---
2 execution:
3   - concurrency: 50
4     hold-for: 3m
5     ramp-up: 2m
6
7   scenario: Choose Flight
8   scenarios:
9     Choose Flight:
10      requests:
11        - label: blazedemo
12          method: GET
13          url: http://blazedemo.com/
14        - label: reserve
15          method: POST
16          url: http://blazedemo.com/reserve.php
17          body:
18            fromPort: Paris
19            toPort: Buenos Aires
```



## Commit changes

Update execution.yml

Add an optional extended description...

- ☒ Commit directly to the `master` branch.
- ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

**Step 13:** Go back to your Jenkins project and you'll see that a new job was triggered automatically from the commit we made at the previous step. Click on the little arrow next to the job and choose '**Console Output**'.

The screenshot shows the Jenkins web interface for a project named 'GithubExample'. On the left, there is a sidebar with navigation links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'GitHub Hook Log'. The main area is titled 'Project GithubExample' and contains links for 'Workspace' and 'Recent Changes'. Below this, there is a 'Permalinks' section. A 'Build History' table shows a single build (#1) from Nov 8, 2018, at 3:50 PM. A dropdown menu is open for this build, showing options: 'Changes', 'Console Output' (highlighted), 'Edit Build Information', 'Polling Log', 'Environment Variables', and 'Git Build Data'.

**Step 14:** You can see that Jenkins was able to pull the Taurus script and run it!

The screenshot shows the 'Console Output' for build #1. The output text is as follows:

```

Started by GitHub push by guysalton21
[EnvInject] - Loading node environment variables.
Building in workspace /var/lib/jenkins/workspace/GithubExample
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/guysalton21/LoadTesting.git # timeout=10
Fetching upstream changes from https://github.com/guysalton21/LoadTesting.git
> git --version # timeout=10
> git fetch --tags --progress https://github.com/guysalton21/LoadTesting.git +refs/heads/*:refs/remotes,
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision baee62e35365fbbde6c7526aleab8beddac7de64 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f baee62e35365fbbde6c7526aleab8beddac7de64
Commit message: "Update execution.yml"
First time build. Skipping changelog.
[GithubExample] $ /bin/bash -xe /tmp/hudson8024230436830700517.sh
+ bzt execution.yml
15:50:44 INFO: Taurus CLI Tool v1.12.1
15:50:44 INFO: Starting with configs: ['execution.yml']
15:50:44 INFO: Configuring...
15:50:44 INFO: Artifacts dir: /var/lib/jenkins/workspace/GithubExample/2018-11-08_15-50-44.260890
15:50:44 INFO: Preparing...
15:50:44 WARNING: There is newer version of Taurus 1.13.1 available, consider upgrading. What's new:

```

# Get Started With Jenkins, GitHub, and BlazeMeter

Congratulations! Every time you publish your changes to Github, GitHub will trigger your new Jenkins job.

Once the GitHub plugin is fully installed and integrated into your Jenkins project, you have completed a very crucial step towards the full CI process. Now, you can proceed to the testing phase. To complete the full CI process, integrate your load testing into your CI tool.

**START TESTING NOW (HTTPS://A.BLAZEMETER.COM/APP/SIGN-UP?\_\_HSTC=46213176.284FBB8EBB81C4E3CC139F4B91BE38F3.1655221191797.1656**

## **Guy Salton (/guy-salton)**

Testing Expert

Guy Salton is a technical expert on the whole performance testing ecosystem - load testing tools, monitoring tools, CI tools, Networking and Infrastructure. His expertise is helping with POCs and special technical projects for strategic customers. Guy talks at conferences and meetups around the world, writes blog posts and gives webinars.

### **PRODUCTS >**

BlazeMeter

(/Product/Blazemeter/)

Functional

(/Product/Blazemeter/)

Testing)

### **SOLUTIONS >**

JMeter

(/Solutions/Jmeter)

Jenkins CI

(/Solutions/Jenkins)

LoadRunner Vs.

BlazeMeter

### **SUPPORT >**

Services

(/Professional-

Services)

Documentation

(Https://Guide.BlazeMeter.com/)

Us?

### **RESOURCES >**

Events & Webinars

(/Resources/Events-

Webinars)

Papers & Videos

(/Resources/Papers-

And-Videos)

### **COMPANY >**

Customers

(/Customers)

News (/Press/Press-

Releases)

**CONTACT >**

SEND FEEDBACK

RELA

LINKS

Produ

Blaze

Load

(Https

Testin

Perfor

Testin

Performance (/Resources/Loadrunner) [Load-Testing-Jmeter- \(/Resources/Recorded-](#)  
 (/Product/Blazemeter/Load-Testing) [Black Friday \(/Solutions/Black-Friday\)](#) [Status \(https://Status.Blazemeter.com\)](#) [Webinars \(/Resources/Recorded-Webinars\)](#) [Videos \(https://www.youtube.com/user/BlazeMeterSupport\)](#)  
 APIs (/Product/Blazemeter/Api) [Changelog \(/Release-Notes/Blazemeter\)](#) [UNIVERSITY >](#)  
 Monitoring) [Mainframe \(/Solutions/Mainframe\)](#) [PRICING >](#) [BLOG >](#)  
 Mock Services (/Product/Blazemeter/Mock-Services)  
 Test Data (/Product/Blazemeter/Test-Data)

Blazemeter by Perforce  
 (https://www.perforce.com)

© 2022 Perforce  
 Software, Inc.

Terms & Conditions  
 (https://www.perforce.com/sites/default/files/pdfs/BlazeMeter-Software-as-a-Service-Agreement\_October-29-2021.pdf) | Privacy Policy  
 (https://www.perforce.com/privacy-policy) | Sitemap  
 (/sitemap)

