



Jenkins

Build Triggers

JENKINS

BY ALOK SRIVASTAVA - NETWORK NUTS



BUILD TRIGGERS

While working with Jenkins, there are different ways we can tell Jenkins to run our projects, Manually. While working on real projects, we need more automated ways to run our builds. Build triggers help to make this possible.



There are different ways we can achieve this:

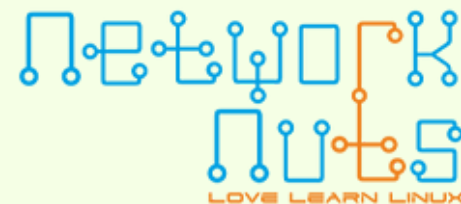
- Starting a build job when another job has been completed using upstream/downstream projects
- Running builds at periodic intervals
- Polling source code management for changes
- Triggering builds remotely

Creating Upstream / Downstream Projects

The first build trigger mechanism mentioned above involves creating two inter-linked build tasks.

We will create two projects and using one to trigger the other.

Two projects, project ONE and project TWO, if project TWO is configured to run once project ONE completes, we call project TWO the downstream project and project ONE the upstream project.




Create project ONE as a simple "parameterized" project with a variable "valone" having default value as "project one".

Enter an item name


ONE

» Required field




Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.




Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

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



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a folder, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

General

Source Code Management

Build Triggers

Build Environment

Build

Post-build Actions

☐ This build requires lockable resources

☒ This project is parameterized

String Parameter

Name

valone

Default Value

project ONE

Description

[Plain text]

[Preview](#)

☐ Trim the string

Add Parameter

☐ Throttle builds

Save

Apply

Build

 Execute shell  

Command

```
echo "$valone"
```

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

Advanced...


Select Apply and Save at the end of the page.

Go back to the dashboard and create Project TWO the same way we set up Project ONE, but with different parameter identifier.

Enter an item name


TWO

» Required field




Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.




Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

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



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

GeneralSource Code ManagementBuild TriggersBuild EnvironmentBuildPost-build Actions

Description

[Plain text] [Preview](#)

☐ Discard old builds

☐ GitHub project

☐ This build requires lockable resources

☒ This project is parameterized

Add Parameter

☐ Throttle builds

☐ Disable this project

?

?

?

?

?

Network
Nites
LOVE LEARN LINUX

☒ This project is parameterized



String Parameter



Name



Default Value



Description



[Plain text] [Preview](#)

☐ Trim the string




Add Parameter ▼

Before we add our build step, we have to configure our project as a downstream project. Under Build Triggers in the project configuration, select Build after other projects are built.

After we select the **Build after other projects are built** option, we are presented with a text field in which we have to enter the upstream project, that is, Project ONE. Start typing the name of the upstream project and Jenkins will autocomplete and filter with the projects that match what you want to specify.

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) 

☒ Build after other projects are built 


Projects to watch


ONE,

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

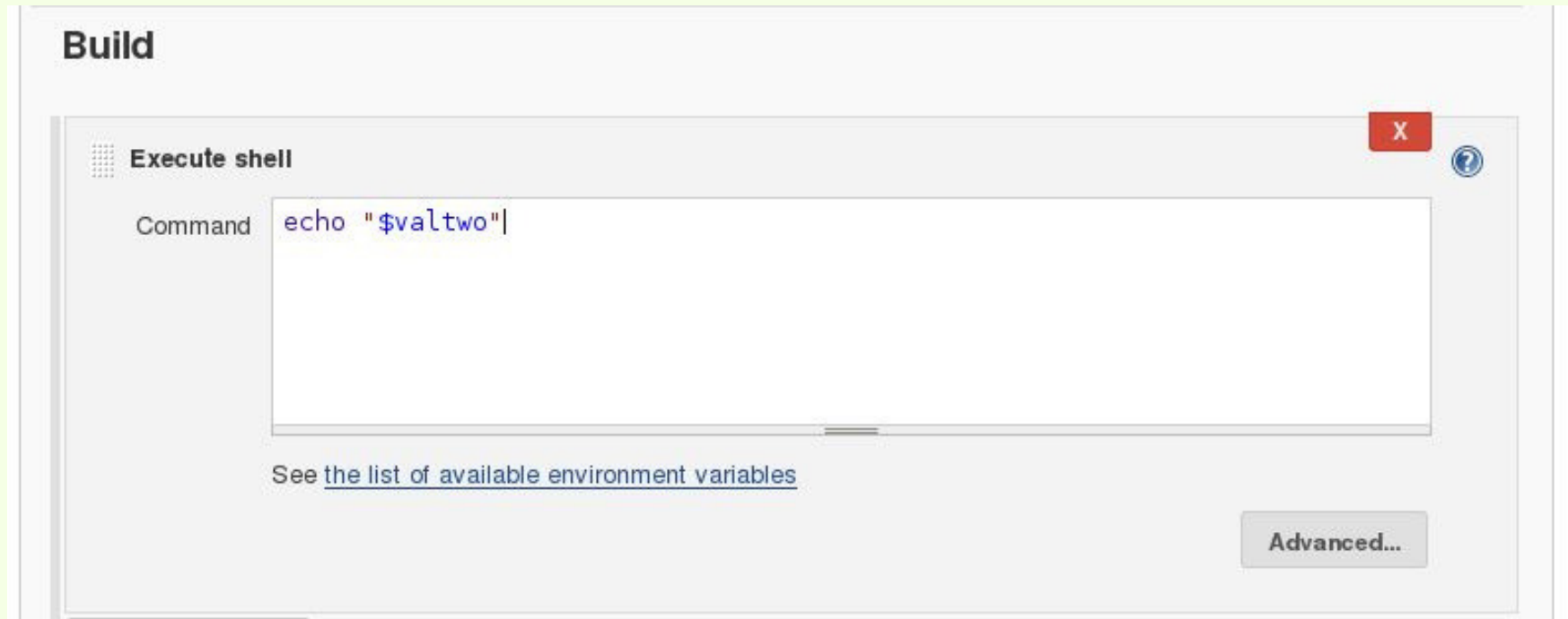
☐ Build periodically 

☐ GitHub hook trigger for GITScm polling 

☐ Poll SCM 

You can configure multiple upstream projects, thus you will notice a comma after Project ONE in the Projects to watch text field.

Finally, add a build step to execute a shell script that will output the parameter we added earlier.



Click **Apply** and **Save** to persist the build configuration you just made.

Before we run our projects, we need to make one more configuration change on our upstream project.

☒ This project is parameterized

String Parameter

Name

valone

Default Value

Welcome to Network Nuts - Jenkins Training

Description

Custom parameter for the lab

[Plain text] [Preview](#)

☐ Trim the string

Add Parameter ▼

While on the Jenkins dashboard, select Project ONE. On the left configuration panel, click on **Configure** and this will open the now familiar project configuration page.

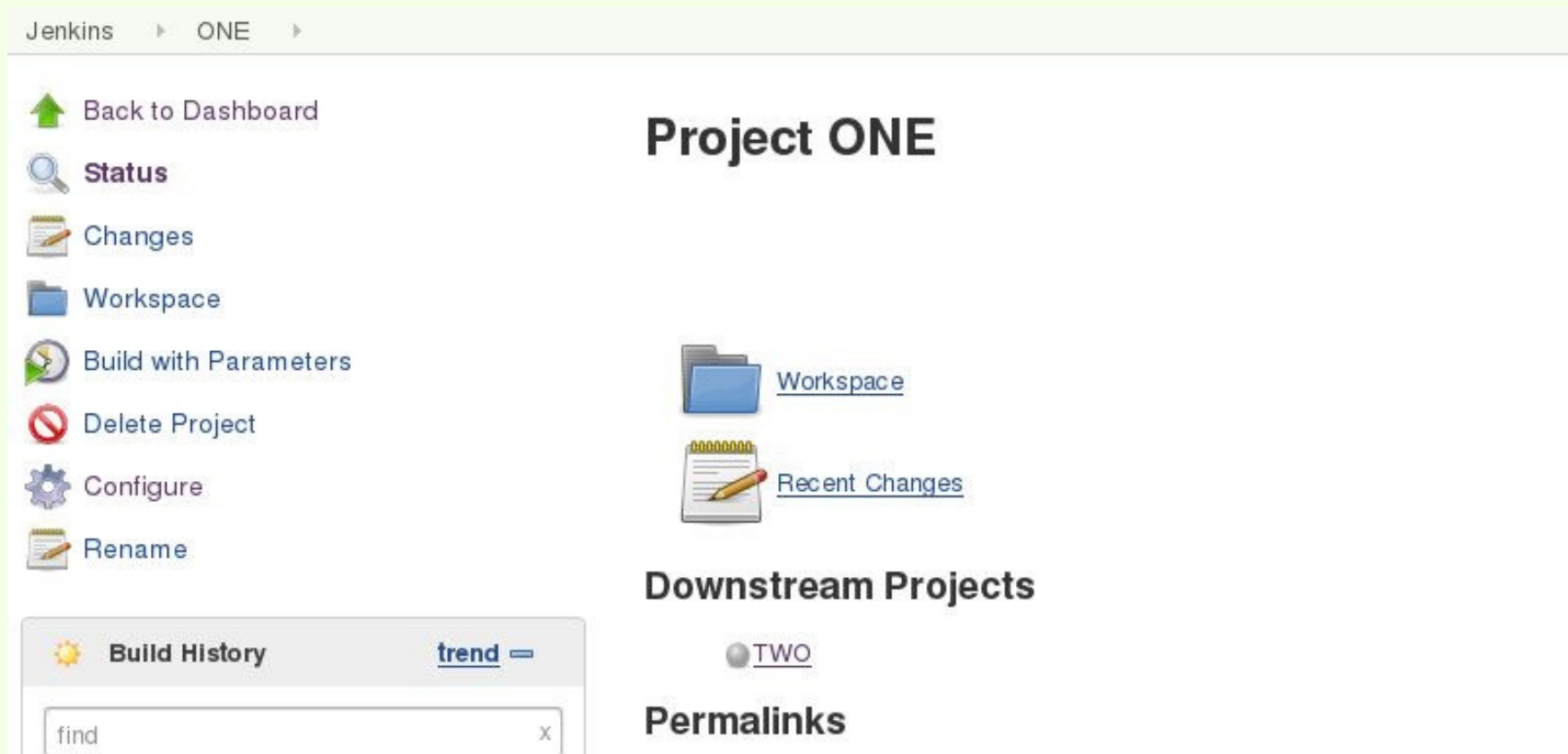
The screenshot shows the Jenkins dashboard with the 'Network Nuts' project selected. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View. The main content area displays a table of builds for the 'Network Nuts' project. The table has columns for Status (S), Weather (W), Name, Last Success, Last Failure, and Last Duration. There are four builds listed: ONE, projectparameter, testgit, and TWO. The 'projectparameter' build shows a success status with a duration of 94 ms. The 'testgit' build shows a success status with a duration of 1.5 sec. The 'ONE' and 'TWO' builds show a failure status with a duration of N/A. Below the table, there is a legend for RSS feeds: RSS for all, RSS for failures, and RSS for just latest builds. The bottom left corner shows the 'Build Queue' section with the message 'No builds in the queue.'

S	W	Name ↓	Last Success	Last Failure	Last Duration
☐	☀	ONE	N/A	N/A	N/A
🔵	☀	projectparameter	43 min - #2	N/A	94 ms
🔵	☀	testgit	1 hr 37 min - #2	N/A	1.5 sec
☐	☀	TWO	N/A	N/A	N/A

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

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

Build Queue
No builds in the queue.



Scroll to the bottom to get to the **Post-build Actions** section. Click on Add post-build action and select **Build other projects**, as shown:

Post-build Actions

Add post-build action ▼

Aggregate downstream test results

Archive the artifacts

Build other projects

Publish JUnit test result report

Record fingerprints of files to track usage

Git Publisher

E-mail Notification

Editable Email Notification

Set GitHub commit status (universal)

Set build status on GitHub commit [deprecated]

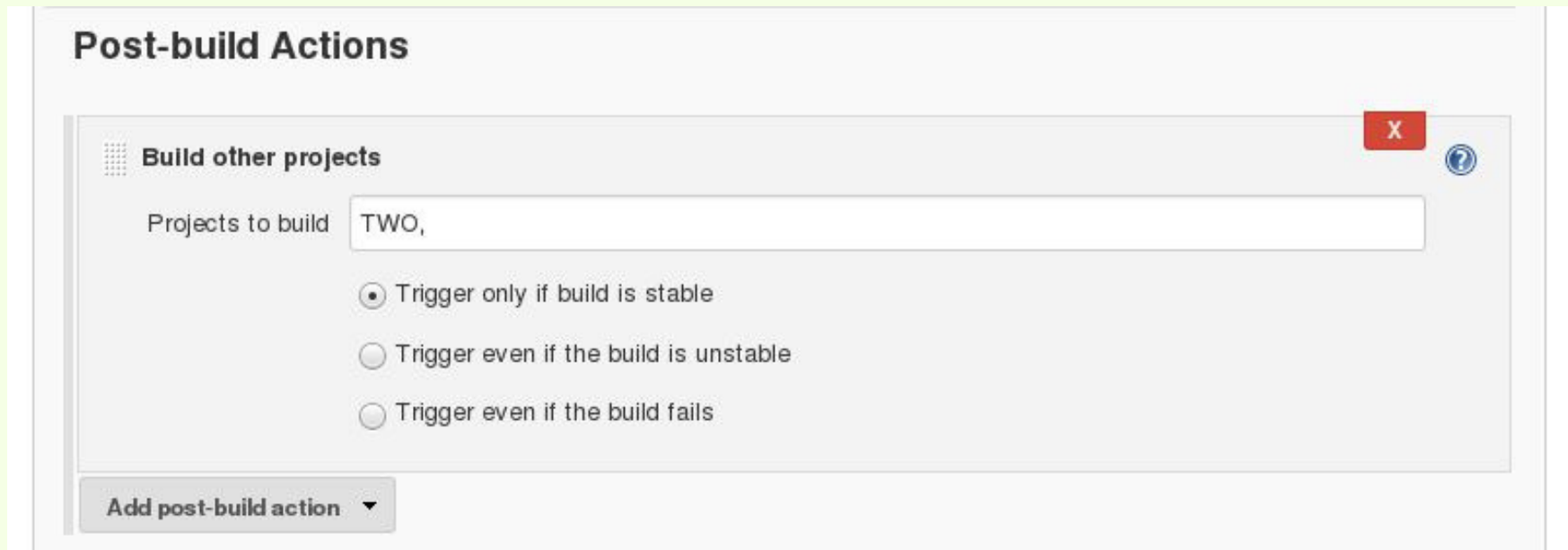
Delete workspace when build is done

[Environment variables](#)

Advanced...

Add post-build action ▼

Enter **Project TWO** in Projects to build and select the Trigger only if build is stable radio button.



The screenshot shows a 'Post-build Actions' configuration panel. Inside, there is a section titled 'Build other projects' with a red 'X' icon and a help icon. Below this, there is a text input field labeled 'Projects to build' containing the text 'TWO,'. Underneath the input field are three radio button options: 'Trigger only if build is stable' (which is selected), 'Trigger even if the build is unstable', and 'Trigger even if the build fails'. At the bottom of the panel is a button labeled 'Add post-build action' with a dropdown arrow.

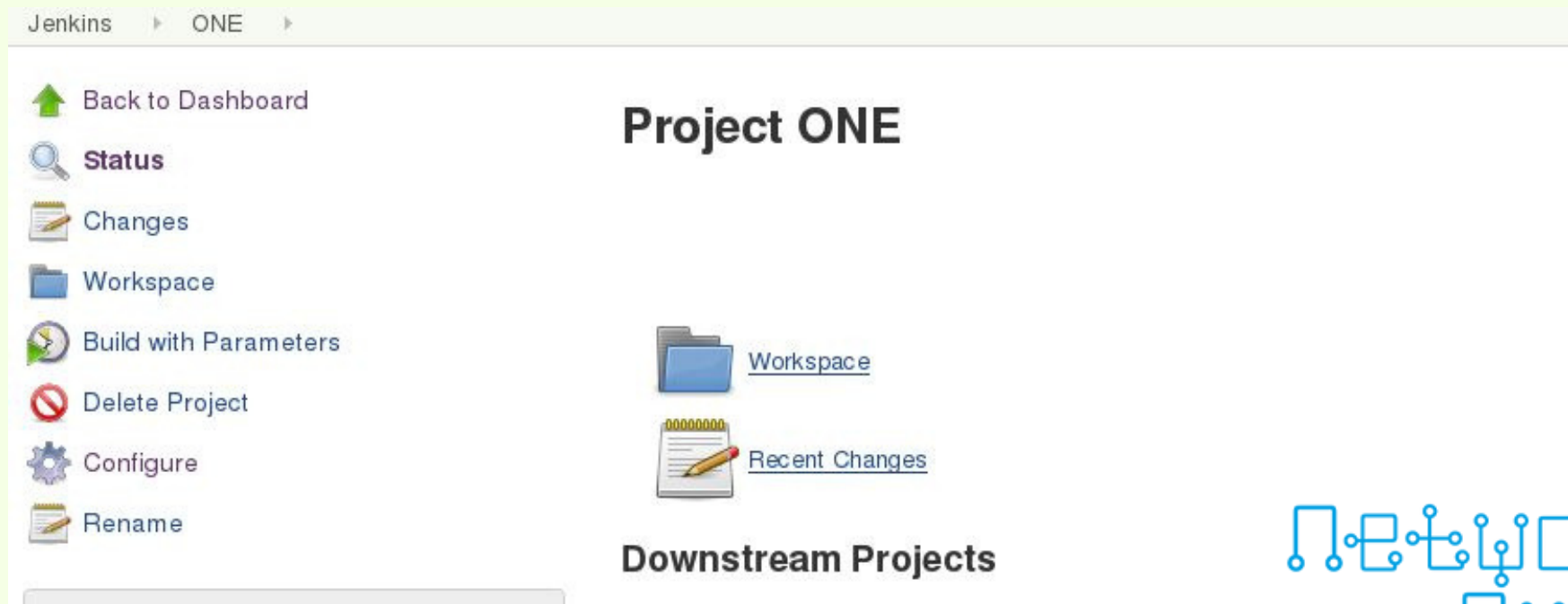
Select **Apply** and **Save** to save the configuration.

At this point, we have completed our upstream/downstream configuration.

Running an Upstream Project

Let's try to run an upstream project in Jenkins.

Jenkins dashboard, click on Project ONE, and, on the left-hand **configuration** panel, select **Build with parameters**. On the Build with Parameters page, click Build.



 [Back to Dashboard](#)


 [Status](#)

 [Changes](#)

 [Workspace](#)

 [Build with Parameters](#)

 [Delete Project](#)

 [Configure](#)


 [Rename](#)

Project ONE

 [Workspace](#)

 [Recent Changes](#)

Downstream Projects

 [TWO](#)

Permalinks

- [Last build \(#1\), 16 ms ago](#)

 **Build History**

[trend](#) 

find


x

 **#1**

Mar 31, 2019 3:48 AM



 [Back to Dashboard](#)


 [Status](#)

 [Changes](#)

 [Workspace](#)

 [Build with Parameters](#)

 [Delete Project](#)

 [Configure](#)

 [Rename](#)



Build History

[trend](#) 

x



#1

Mar 31, 2019 3:48 AM



[RSS for all](#)



[RSS for failures](#)

Project ONE



[Workspace](#)



[Recent Changes](#)

Downstream Projects



[TWO](#)

Permalinks

- [Last build \(#1\), 20 sec ago](#)
- [Last stable build \(#1\), 20 sec ago](#)
- [Last successful build \(#1\), 20 sec ago](#)
- [Last completed build \(#1\), 20 sec ago](#)

Select Project TWO under the Downstream Projects section. While on Project TWO, go the Build History section on the left and hover over the latest build, click the down arrow and select Console Output.

The screenshot shows the Jenkins web interface for 'Project TWO'. The breadcrumb navigation at the top reads 'Jenkins > TWO >'. On the left sidebar, there are links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build with Parameters', 'Delete Project', 'Configure', and 'Rename'. The 'Build History' section is expanded, showing a table with one build: '#1' on 'Mar 31, 2019 3:49 AM'. Below the table are links for 'RSS for all' and 'RSS for failures'. On the right, the 'Project TWO' header is followed by links for 'Workspace' and 'Recent Changes'. Below that is the 'Upstream Projects' section showing 'ONE'. The 'Permalinks' section lists four links: 'Last build (#1), 48 sec ago', 'Last stable build (#1), 48 sec ago', 'Last successful build (#1), 48 sec ago', and 'Last completed build (#1), 48 sec ago'.

Jenkins > TWO >

Project TWO

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build with Parameters](#)

[Delete Project](#)

[Configure](#)

[Rename](#)

Build History [trend](#)

find x

[#1](#) Mar 31, 2019 3:49 AM

[RSS for all](#) [RSS for failures](#)

[Workspace](#)

[Recent Changes](#)

Upstream Projects

[ONE](#)

Permalinks

- [Last build \(#1\), 48 sec ago](#)
- [Last stable build \(#1\), 48 sec ago](#)
- [Last successful build \(#1\), 48 sec ago](#)
- [Last completed build \(#1\), 48 sec ago](#)

Looking at the output, we can see that the build of Project TWO was triggered by its upstream project. The output also informs us that Project ONE was run by the user.



The screenshot shows the Jenkins web interface for a build of Project TWO. The breadcrumb navigation at the top reads 'Jenkins > TWO > #1'. On the left sidebar, there are links for 'Back to Project', 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', 'Delete build '#1'', and 'Parameters'. The main content area is titled 'Console Output' with a blue sphere icon. The output text shows the build was started by an upstream project 'ONE' build number 1, originally caused by user 'alok.srivastava'. It then shows the build running in the workspace, executing a shell script that prints 'project two', and finally finishing with 'SUCCESS'.

Jenkins > TWO > #1

- Back to Project
- Status
- Changes
- Console Output**
- View as plain text
- Edit Build Information
- Delete build '#1'
- Parameters

Console Output

```
Started by upstream project "ONE" build number 1
originally caused by:
  Started by user alok.srivastava
Started by upstream project "ONE" build number 1
originally caused by:
  Started by user alok.srivastava
Building in workspace /var/lib/jenkins/workspace/TWO
[TWO] $ /bin/sh -xe /tmp/jenkins6467378853564563633.sh
+ echo 'project two'
project two
Finished: SUCCESS
```