

Automating Your Automation

The Care and Feeding of Jenkins

Jeff McKenzie • jeff.mckenzie@insight.com • [@jeffreymckenzie](https://twitter.com/jeffreymckenzie)

Good afternoon everyone –
Thanks for being here.....

We're going to be looking at Jenkins
-- brief intro
-- how to care for it

My name is Jeff McKenzie,
And I am a Practice Manager for App Dev and Infrastructure
At Insight Digital Innovation in Columbus Ohio

@jeffreymckenzie



[How many Jenkins, types of jobs?]

If you're interested in automation,

You want to interact with Jenkins as little as possible.

So we are going to be looking at a case study

To demonstrate ways we can scale Jenkins in a better way.

And when I say case study....

=====

Logo from <https://jenkins.io/artwork>

@jeffreymckenzie



I'm talking about Santa

--case study of all case studies

-- you talk about someone who needs to scale....

Specifically, communicate with his reindeer

So let's look at an example of what Santa's going to do here.

-- create new freestyle job called Rudolph...

=====

Logo from <https://jenkins.io/artwork>

Created by https://twitter.com/ks_nenasheva

@jeffreymckenzie

New Item [Jenkins] x +


localhost:8080/view/all/newJob

Jenkins Jeff McKenzie | log out


Jenkins > All >

Enter an item name


» 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**

For projects that need a large number of different configurations, such as testing

OK

Click OK...

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'rudolph'. The browser address bar indicates the URL is 'localhost:8080/job/rudolph/configure'. The Jenkins logo and user 'Jeff McKenzie' are visible in the top navigation bar. The configuration page has several tabs: 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. The 'General' tab is active. It contains a 'Description' field with the text 'This freestyle job says hello to Rudolph.' Below this is a '[Plain text] Preview' link. There is a checked checkbox for 'Discard old builds'. Under this, the 'Strategy' is set to 'Log Rotation'. There are two input fields: 'Days to keep builds' (empty) and 'Max # of builds to keep' (set to '3'). Explanatory text states that build records are kept up to the specified number of days or the maximum number of builds, whichever is less. At the bottom left are 'Save' and 'Apply' buttons, and at the bottom right is an 'Advanced...' link.

rudolph Config [Jenkins]

localhost:8080/job/rudolph/configure

Jenkins

Jeff McKenzie | log out

Jenkins > rudolph >

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

Description This freestyle job says hello to Rudolph.

[Plain text] [Preview](#)

☒ Discard old builds

Strategy Log Rotation

Days to keep builds

if not empty, build records are only kept up to this number of days

Max # of builds to keep 3

if not empty, only up to this number of build records are kept

Save Apply Advanced...

-- description, says hello to Rudolph

-- keep last 3 builds

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'rudolph'. The 'Source Code Management' tab is selected, showing options for 'None' and 'Git'. The 'Git' option is chosen. Below this, the 'Repositories' section contains a form with a 'Repository URL' field, a 'Credentials' dropdown set to '- none -', and an 'Add' button. A red error message 'Please enter Git repository.' is displayed above the 'Add' button. To the right of the 'Add' button are 'Advanced...' and 'Add Repository' buttons. At the bottom, the 'Branches to build' section has a 'Branch Specifier (blank for 'any')' field containing '*/master' and an 'Add Branch' button. A 'Save' button is visible in the bottom left corner of the configuration area.

Under the source code management tab, select Git

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'rudolph'. The 'Source Code Management' tab is selected. Under the 'Source Code Management' section, 'Git' is chosen as the provider. In the 'Repositories' section, the 'Repository URL' is set to 'file:///Users/jmckenzie/projects/santa'. The 'Credentials' dropdown is set to '- none -'. Below this, there are buttons for 'Advanced...', 'Add Repository', and 'Add Branch'. In the 'Branches to build' section, the 'Branch Specifier (blank for 'any')' is set to '*/master'. At the bottom left, there are 'Save' and 'Apply' buttons.

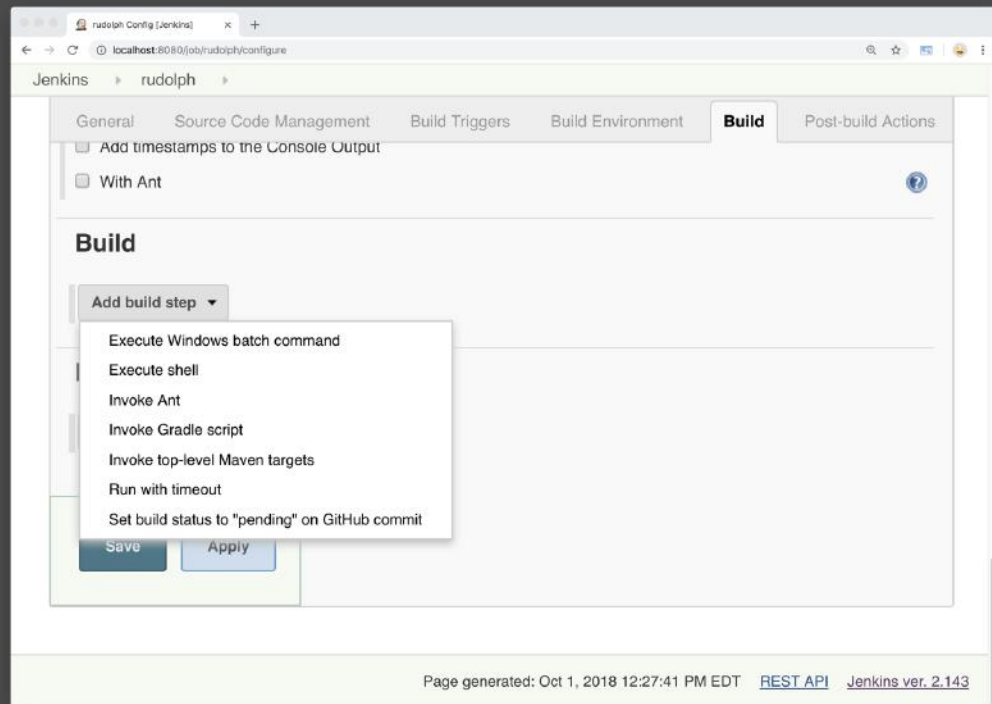
I'm just going to point this to a local git repo

That I have in my home directory

And it's pulling from the master branch

[Git users here?]

@jeffreymckenzie



On the build tab, I'm going to choose

The Execute shell build step...

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'rudolph'. The browser address bar indicates the URL is `localhost:8080/job/rudolph/configure`. The configuration page has several tabs: 'General', 'Source Code Management', 'Build Triggers', 'Build Environment' (which is currently selected), and 'Build'. Under the 'Build Environment' tab, there is a 'Post-build Actions' section with two checkboxes: 'Add timestamps to the Console Output' and 'With Ant', both of which are currently unchecked. Below this is a 'Build' section containing an 'Execute shell' step. This step has a 'Command' text area that is currently empty. Below the text area is a link that says 'See the list of available environment variables'. To the right of the 'Execute shell' step header is a red 'X' icon and a help icon. At the bottom right of the 'Execute shell' step is an 'Advanced...' button. At the very bottom of the configuration page are two buttons: 'Save' and 'Apply'.

rudolph Config [Jenkins]

localhost:8080/job/rudolph/configure

Jenkins » rudolph »

General Source Code Management Build Triggers **Build Environment** Build

Post-build Actions

☐ Add timestamps to the Console Output

☐ With Ant

Build

Execute shell

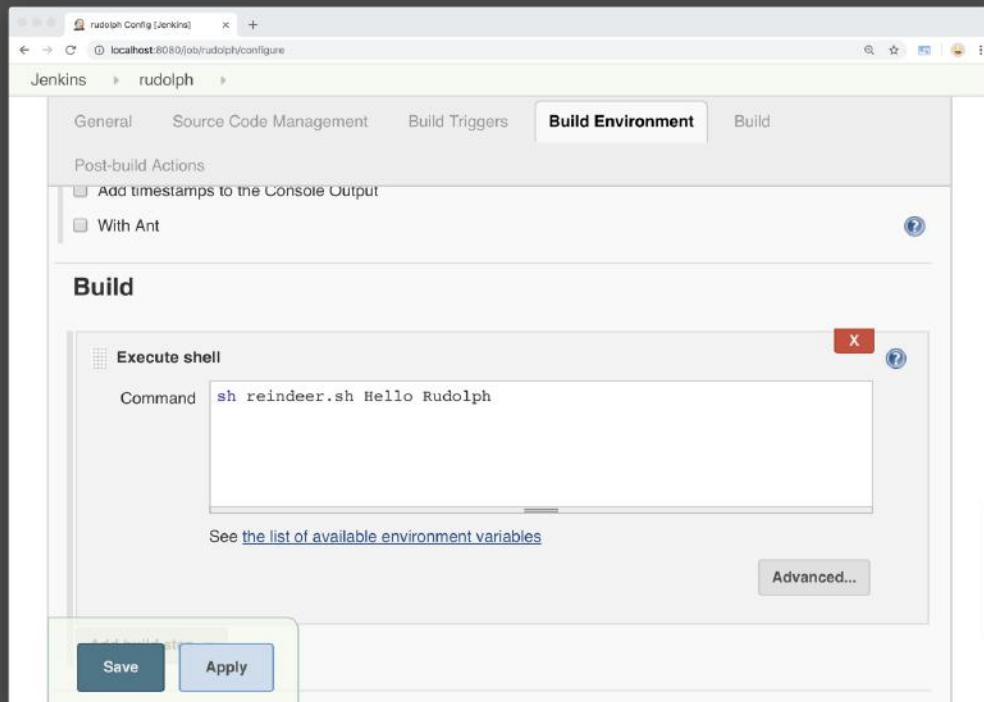
Command

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

Advanced...

Save Apply

@jeffreymckenzie



And put the following command in there...

Sh Reindeer.sh Hello Rudolph.

This tells Jenkins to run a shell script called reindeer.sh

That's in my git repo, so let's take a look at that real quick.

```
#!/bin/bash
greeting="$1"
reindeer_name="$2"
if [ -z "$greeting" ]
then
    greeting="Hi"
fi
if [ -z "$reindeer_name" ]
then
    reindeer_name="Reindeer"
fi
echo =====
echo =
echo = ${greeting} ${reindeer_name}!
echo =
echo =====
```

So this sets two variables –
Greeting, and reindeer name,
Which get passed in from Jenkins.
If nothing gets passed in for either variable,
Then the script sets defaults.
Then the result gets written as output.

So we will save this, go back to the project screen

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a project named 'rudolph'. The browser address bar indicates the URL is `localhost:8080/job/rudolph/`. The Jenkins logo and a search bar are at the top. A user 'Jeff McKenzie' is logged in, with a 'log out' link. A left sidebar contains navigation links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project rudolph' and includes the text 'This freestyle job says hello to Rudolph.' Below this are links for 'Workspace' and 'Recent Changes'. On the right, there are links for 'edit description' and a 'Disable Project' button. A 'Build History' section at the bottom left has a 'trend' dropdown and a search box. At the bottom right, there are 'Permalinks' and RSS feeds for 'all' and 'failures'. The footer shows the page was generated on 'Oct 1, 2018 1:21:43 PM EDT' and provides links for 'REST API' and 'Jenkins ver. 2.143'.

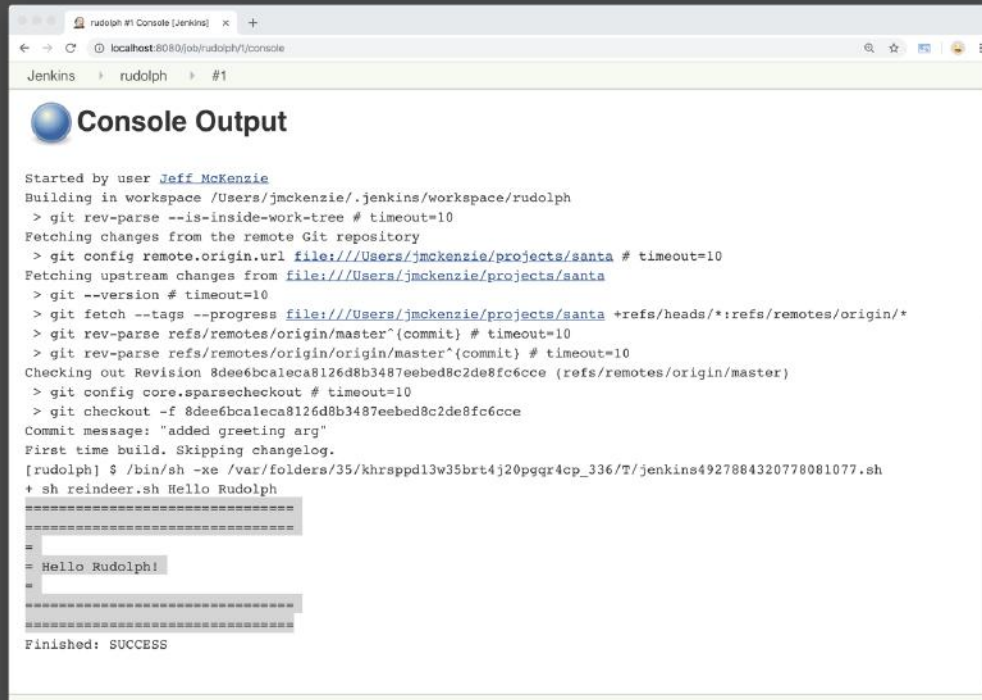
And click “build Now”

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a project named 'rudolph'. The browser address bar indicates the URL is `localhost:8080/job/rudolph/`. The Jenkins header includes a search bar, the user 'Jeff McKenzie', and a 'log out' link. A left sidebar contains navigation links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project rudolph' and contains the text 'This freestyle job says hello to Rudolph.' Below this are links for 'Workspace' and 'Recent Changes'. On the right, there are links for 'edit description' and a 'Disable Project' button. A 'Build History' section shows a single build from 'Oct 1, 2018 1:23 PM' with a status of '#1'. At the bottom, there are 'RSS for all' and 'RSS for failures' links. The footer indicates the page was generated on 'Oct 1, 2018 1:21:43 PM EDT' and provides links for 'REST API' and 'Jenkins ver. 2.143'.

And we have a success – let's look at the output.

@jeffreymckenzie



The screenshot shows a web browser window displaying the Jenkins console output for a job named 'rudolph'. The output text is as follows:

```
Started by user Jeff_McKenzie
Building in workspace /Users/jmckenzie/.jenkins/workspace/rudolph
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url file:///Users/jmckenzie/projects/santa # timeout=10
Fetching upstream changes from file:///Users/jmckenzie/projects/santa
> git --version # timeout=10
> git fetch --tags --progress file:///Users/jmckenzie/projects/santa +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 8dee6bca8126d8b3487eebed8c2de8fc6cce (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 8dee6bca8126d8b3487eebed8c2de8fc6cce
Commit message: "added greeting arg"
First time build. Skipping changelog.
[rudolph] $ /bin/sh -xe /var/folders/35/khrsppdl3w35brt4j20pgqr4cp_336/T/jenkins4927884320778081077.sh
+ sh reindeer.sh Hello Rudolph
=====
= Hello Rudolph!
=====
Finished: SUCCESS
```

There we have the result of the script as it's run.
So there's one job for one reindeer.

- contrived example, but basic
- does have the basic elements:
 - pulling code from source control
 - and doing something with it
- not concerned as much with what it does as
 - how to scale and manage these types of jobs

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a job named 'Project rudolph'. The browser address bar shows 'localhost:8080/job/rudolph/'. The Jenkins logo is in the top left, and the user 'Jeff McKenzie' is logged in. The left sidebar contains navigation links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project rudolph' and includes a description: 'This freestyle job says hello to Rudolph.' Below this are links for 'Workspace' and 'Recent Changes'. A 'Build History' section shows a single build (#1) from Oct 1, 2018 at 1:23 PM. A 'Permalinks' section lists links for the last build, last stable build, last successful build, and last completed build, all from 19 hours ago. At the bottom, there are links for 'RSS for all' and 'RSS for failures'. The footer indicates the page was generated on Oct 2, 2018 at 9:22:47 AM EDT, with links for 'REST API' and 'Jenkins ver. 2.143'.

Here's our Rudolph project.

Let's say we copy this job 9 times for a total of 10 reindeer,
And we want this to say something other than Hello as a greeting?

[change each one individually....]

@jeffreymckenzie



You get spaghetti automation that ties your hands together.

There's a better way to do this....

=====

[https://commons.wikimedia.org/wiki/File:Homemade_fresh_pasta_\(spaghetti\)_by_Camille_\(Unsplash\).jpg](https://commons.wikimedia.org/wiki/File:Homemade_fresh_pasta_(spaghetti)_by_Camille_(Unsplash).jpg)

By davide ragusa davideragusa (<https://unsplash.com/photos/FwiLgvi-2Do>) [CC0], via Wikimedia Commons

@jeffreymckenzie

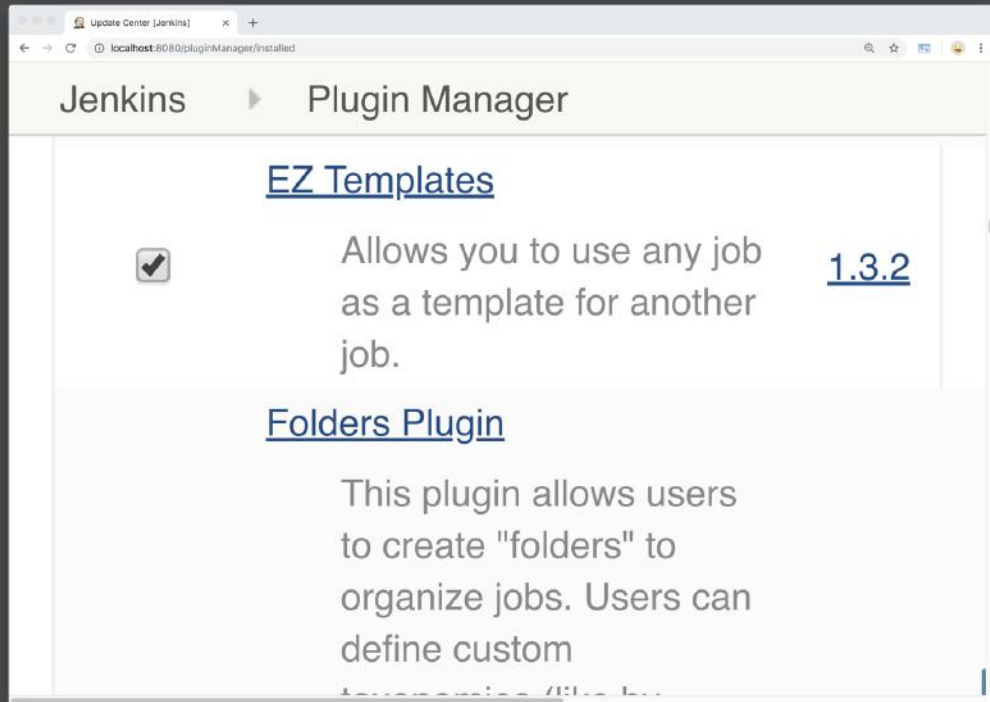
The screenshot shows the Jenkins Plugins page for the 'EZ Templates' plugin. The page is titled 'EZ Templates 1.3.2' and includes the following information:

- Minimum Jenkins requirement:** 1.642.3
- ID:** ez-templates
- Installs:** 2898
- Last released:** 9 months ago
- Maintainers:** Marc Carter, Joel Johnson
- Dependencies:** promoted builds v.2.21 (optional), bouncycastle API v.2.16.0 (implied), Command Agent Launcher v.1.0 (implied), JDK Tool v.1.0 (implied)
- Description:** Allows you to use any job as a template for other jobs.
- Usage:** See the [GitHub home](#) for this plugin.
- Changelog:** 1.3.2
 - Feature: JENKINS-45257 EZ-Templates must rollback any error while applying template to a single

On the right side of the page, there is an 'Archives' section with a graph showing the number of installations over time, and a 'Labels' section with the label 'misc'. At the bottom right, there is a section titled 'ARE YOU MAINTAINING THIS PLUGIN?' with a link to the Jenkins Plugin Wiki.

And one way is to use a plugin called EZ Templates
Here's the official plugin page for it,
There's also a git repo for it
the plugin itself is available in the Manage Plugins
section of Jenkins

@jeffreymckenzie



So you just need to select and install it

As it describes, this allows you to take any job
And use it as a template for other jobs.

Let's take a look at how it might help us scale in this instance.

@jeffreymckenzie

New Item [Jenkins]

localhost:8080/view/Jenkins/newJob


Jenkins


Jeff McKenzie | log out


Enter an item name

reindeer-template

» 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.

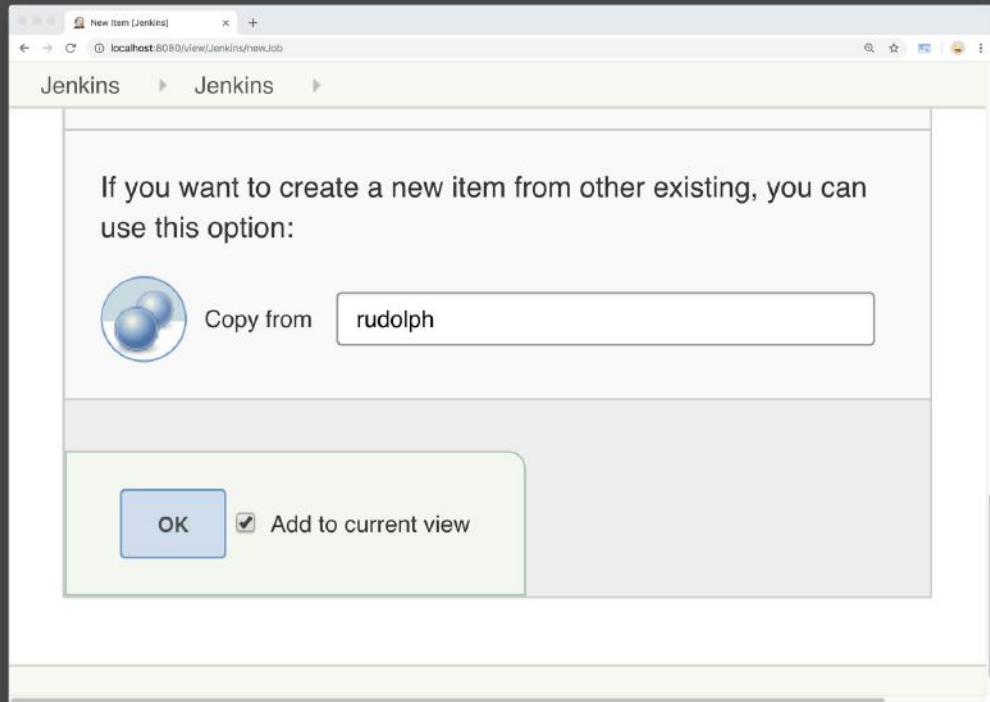
 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Organizes a large number of different configurations, such as testing

OK ☒ Add to current view

Create new freestyle job
Let's call it Reindeer Template...

@jeffreymckenzie



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/view/Jenkins/newJob'. The page title is 'New Item [Jenkins]'. The main content area has a heading 'If you want to create a new item from other existing, you can use this option:' followed by a blue circular icon with three spheres. To the right of the icon is the text 'Copy from' and a text input field containing the value 'rudolph'. At the bottom left, there is a blue 'OK' button. To its right is a checked checkbox followed by the text 'Add to current view'.

And we will create it as a copy of Rudolph

So we can use that as a starting point.

@jeffreymckenzie

The screenshot shows the Jenkins web interface. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/reindeer-template/configure`. The Jenkins logo and user name 'Jeff McKenzie' are visible in the top navigation bar. The breadcrumb trail shows 'Jenkins > Jenkins > reindeer-template'. The 'General' tab is selected, showing the job's configuration. The 'Description' field contains the text 'This freestyle job says hello to Rudolph.' Below it, there is a '[Plain text] Preview' link. The 'Discard old builds' checkbox is checked. The 'Strategy' dropdown is set to 'Log Rotation'. The 'Days to keep builds' field is empty, with a note: 'If not empty, build records are only kept up to this number of days'. The 'Max # of builds to keep' field is set to '3', with a note: 'if not empty, only up to this number of build records are kept'. At the bottom left, there are 'Save' and 'Apply' buttons.

reindeer-template Config [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-template/configure

Jenkins

Jeff McKenzie | log out

Jenkins > Jenkins > reindeer-template

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

Description

This freestyle job says hello to Rudolph.

[Plain text] [Preview](#)

☒ Discard old builds

Strategy Log Rotation

Days to keep builds

If not empty, build records are only kept up to this number of days

Max # of builds to keep 3

if not empty, only up to this number of build records are kept

Save Apply

Let's go ahead and change the description
So it makes sense as a template

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'reindeer-template'. The browser address bar shows 'localhost:8080/view/Jenkins/job/reindeer-template/configure'. The Jenkins logo and user 'Jeff McKenzie' are at the top. The configuration tabs include 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. The 'General' tab is active, showing a 'Description' field with the text 'Serves as a template for the Reindeer jobs.' and a '[Plain text] Preview' link. Below this, the 'Discard old builds' checkbox is checked. The 'Strategy' dropdown is set to 'Log Rotation'. The 'Days to keep builds' field is empty, with a note: 'If not empty, build records are only kept up to this number of days'. The 'Max # of builds to keep' field contains the value '3', with a note: 'if not empty, only up to this number of build records are kept'. At the bottom left are 'Save' and 'Apply' buttons.

reindeer-template Config [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-template/configure

Jenkins

Jeff McKenzie | log out

Jenkins > Jenkins > reindeer-template

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

Description

Serves as a template for the Reindeer jobs.

[Plain text] [Preview](#)

☒ Discard old builds

Strategy Log Rotation

Days to keep builds

If not empty, build records are only kept up to this number of days

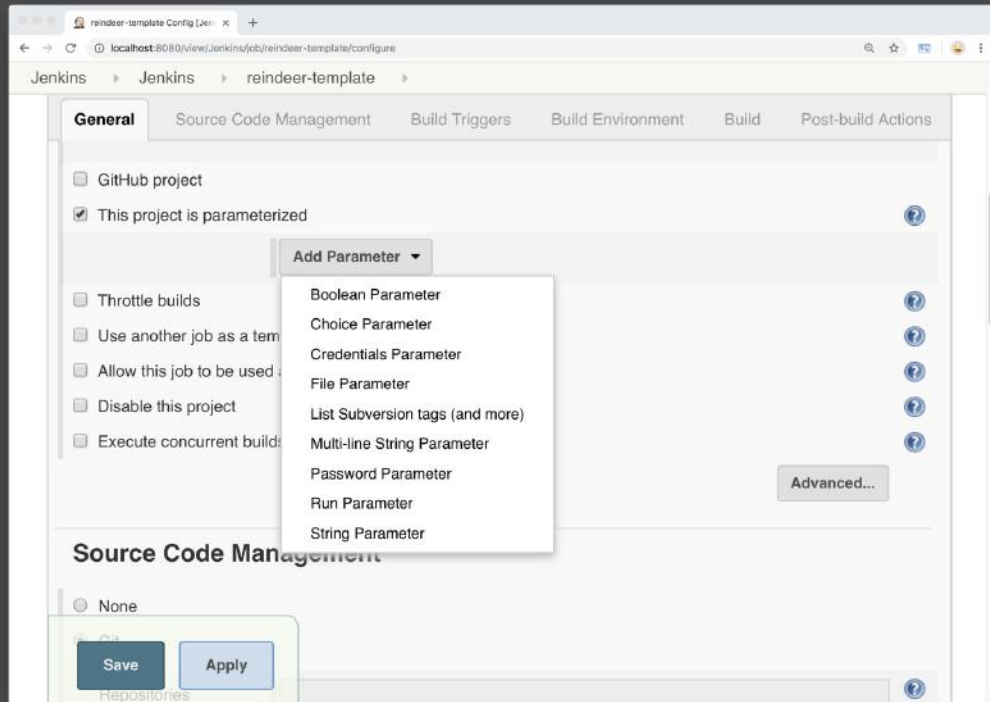
Max # of builds to keep 3

if not empty, only up to this number of build records are kept

Save Apply

“Serves as a template for the reindeer jobs”

@jeffreymckenzie



We are also going to add a couple of parameters (variables)
So Santa can personalize his reindeer greeting –

Check box for “project is parameterized”

And we will select the string parameter

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'reindeer-template'. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/reindeer-template/configure`. The 'General' tab is selected, showing options for 'GitHub project' and 'This project is parameterized' (checked). A 'String Parameter' configuration form is visible with fields for 'Name', 'Default Value', and 'Description'. The 'Description' field is a text area. Below the text area, there is a '[Plain text] Preview' link and a 'Trim the string' checkbox. At the bottom left, 'Save' and 'Apply' buttons are highlighted with a yellow box. An 'Add Parameter' dropdown button is located at the bottom center of the parameter configuration area.

reindeer-template Config [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-template/configure

Jenkins > Jenkins > reindeer-template

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

☐ GitHub project

☒ This project is parameterized

String Parameter

Name

Default Value

Description

[Plain text] [Preview](#)

☐ Trim the string

Add Parameter

Save Apply

Let's enter our values...

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'reindeer-template'. The 'General' tab is selected, and the 'This project is parameterized' checkbox is checked. A 'String Parameter' is configured with the following details:

- Name:** GREETING
- Default Value:** Hello
- Description:** How to say Hello to the reindeer.
- Text Type:** [Plain text] [Preview](#)
- Trim the string:** ☐

At the bottom of the configuration, there are 'Save' and 'Apply' buttons, and an 'Add Parameter' dropdown menu.

We'll call this parameter GREETING
With a default value of Hello
And also give it a description.

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'reindeer-template'. The 'General' tab is active, displaying a 'String Parameter' configuration. The parameter name is 'REINDEER', the default value is empty, and the description is 'The reindeer to say Hello to.'.

String Parameter

Name: REINDEER

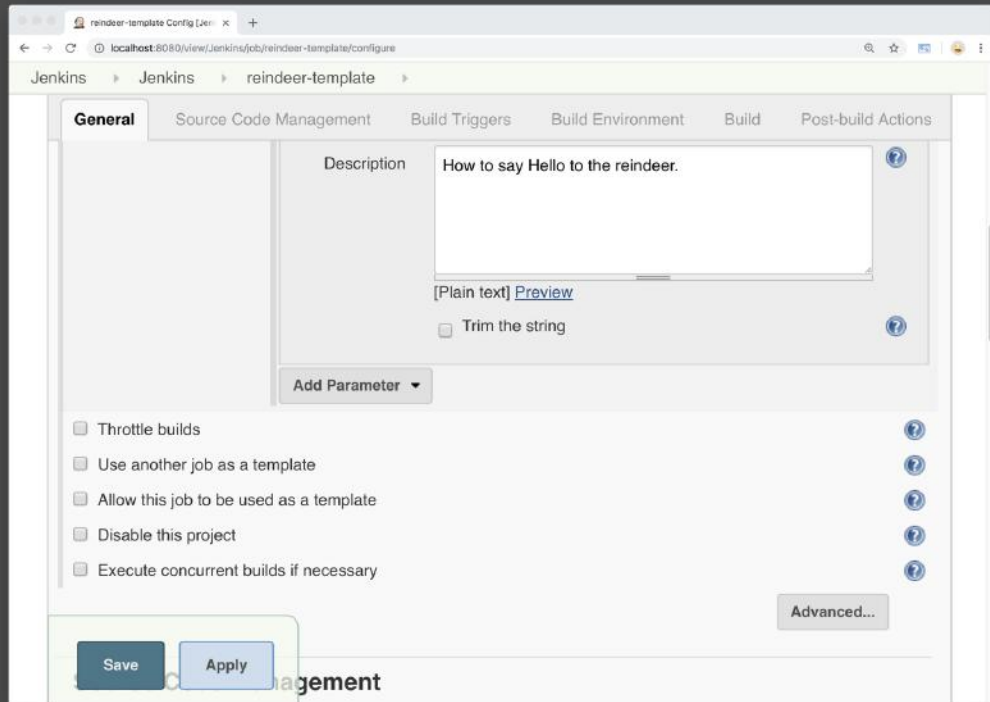
Default Value:

Description: The reindeer to say Hello to.

Buttons: Save, Apply, Add Parameter

Next we will add parameter REINDEER
With no default value
And also give it a description –
“The reindeer to say hello to”

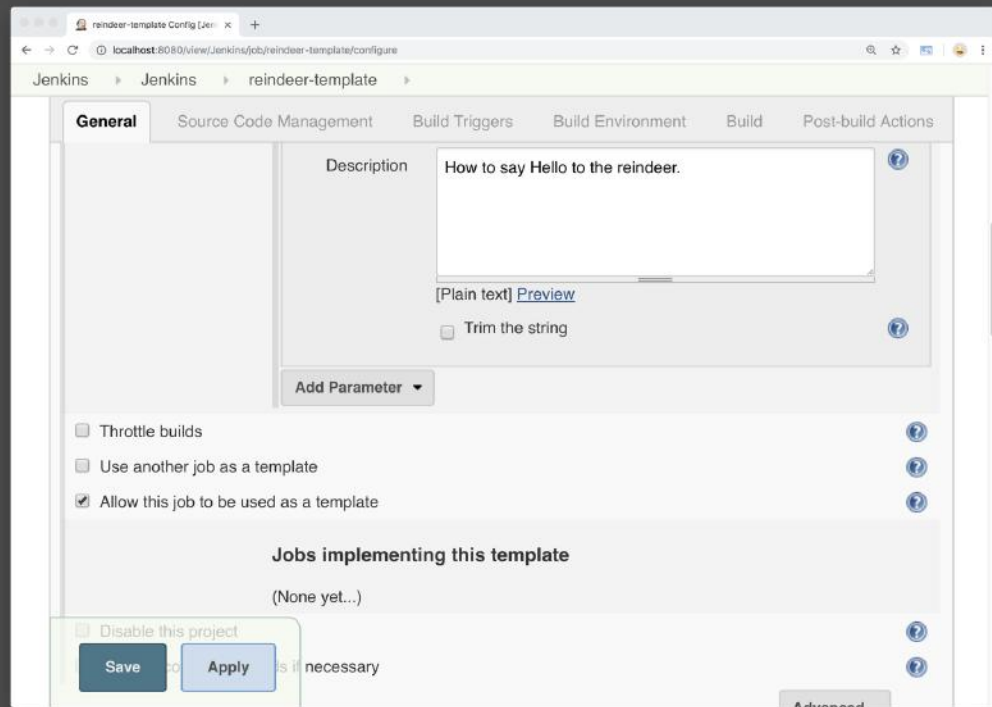
@jeffreymckenzie



Next we will scroll down to the checkbox that says
“Allow this job to be used as a template”

[you see this option only if you have EZ Templates installed]

@jeffreymckenzie



so we check that...

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'reindeer-template'. The 'Source Code Management' tab is selected, showing options for 'None' and 'Git'. The 'Git' option is chosen. Under 'Repositories', the 'Repository URL' is set to 'file:///Users/jmckenzie/projects/santa' and the 'Credentials' are set to '- none -'. There are buttons for 'Advanced...', 'Add Repository', and 'Add Branch'. Under 'Branches to build', the 'Branch Specifier (blank for 'any')' is set to '*/master'. There are 'Save' and 'Apply' buttons at the bottom left.

reindeer-template Config [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-template/configure

Jenkins > Jenkins > reindeer-template >

General **Source Code Management** Build Triggers Build Environment Build

Post-build Actions

Source Code Management

☐ None
☒ Git

Repositories

Repository URL

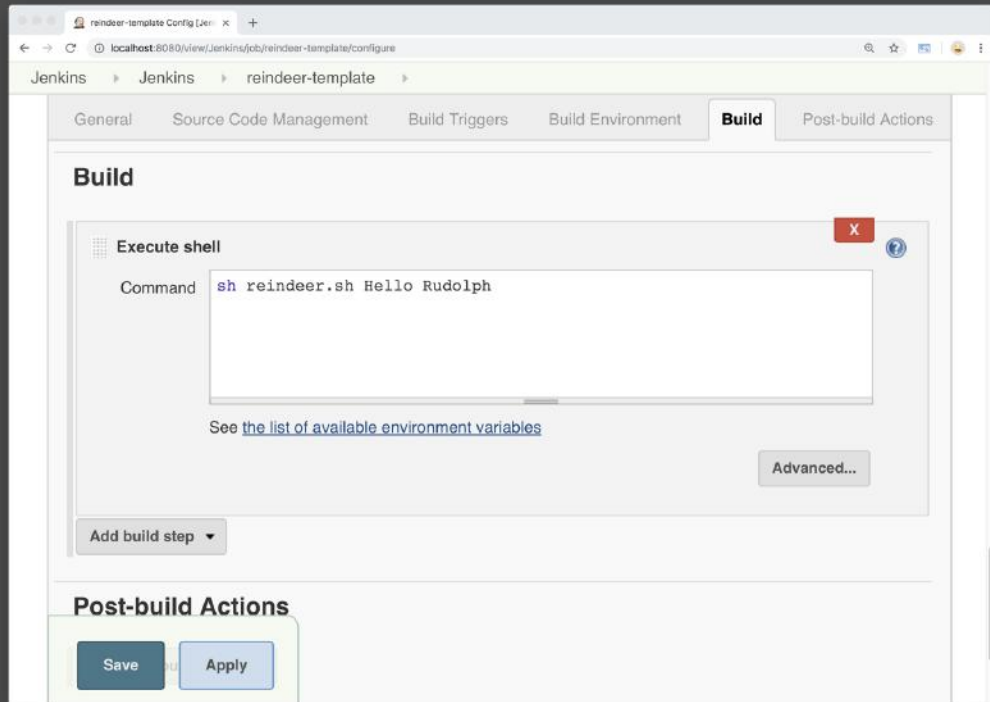
Credentials

Branches to build

Branch Specifier (blank for 'any')

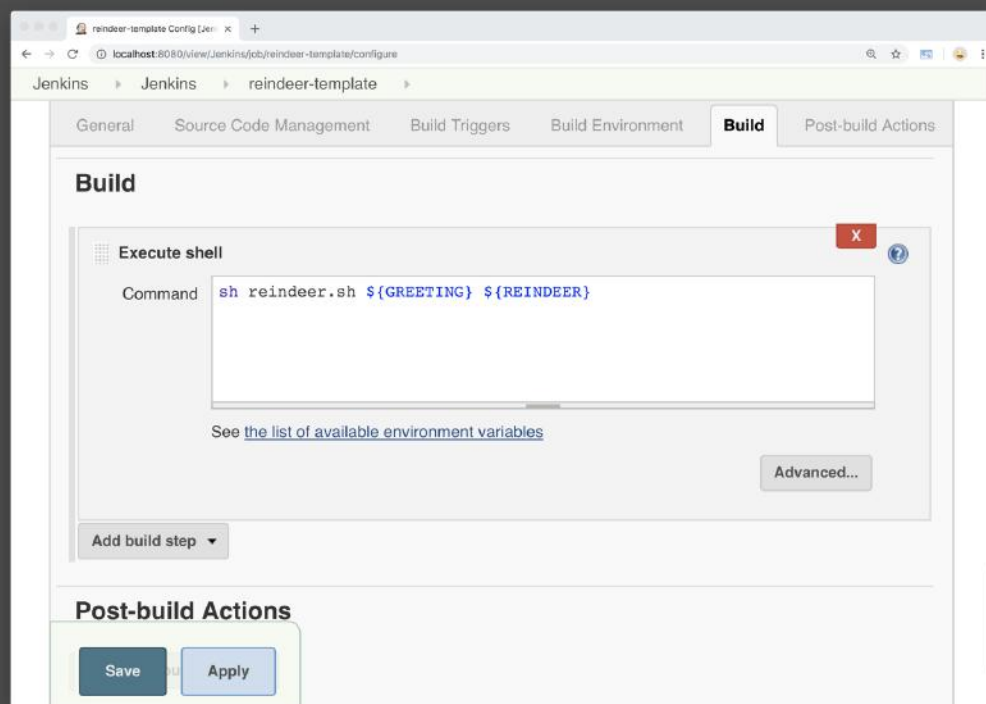
We leave our git repo as it is....

@jeffreymckenzie



We will also change our shell command
To use the parameters

@jeffreymckenzie



Then we will click save...

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a project named 'reindeer-template'. The browser address bar shows 'localhost:8080/view/Jenkins/job/reindeer-template/'. The Jenkins logo and a search bar are at the top. A navigation sidebar on the left includes links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build with Parameters', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project reindeer-template' and contains the text 'Serves as a template for the Reindeer jobs.' and 'ez-templates: This job is configured as a template'. There are links for 'edit description' and a 'Disable Project' button. Below this, there are links for 'Workspace' and 'Recent Changes'. A 'Build History' section with a 'trend' button and a search input is also visible. At the bottom, there are 'Permalinks' for 'RSS for all' and 'RSS for failures'. The footer indicates the page was generated on Oct 2, 2018, at 11:21:28 AM EDT, with links for 'REST API' and 'Jenkins ver. 2.143'.

Which takes us back to the project page

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a project named 'reindeer-template'. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/reindeer-template/`. The page title is 'Project reindeer-template'. Below the title, it says 'Serves as a template for the Reindeer jobs.' There is a green puzzle piece icon and a message: 'ez-templates: This job is configured as a template'. To the right of this message is a link 'edit description'. Below the message is a blue button labeled 'Disable Project'. Further down, there are two links: 'Workspace' (with a folder icon) and 'Recent Changes' (with a notepad icon). At the bottom of the page, it says 'Page generated: Oct 2, 2018 11:21:28 AM EDT' followed by links for 'REST API' and 'Jenkins ver. 2.143'.

Because this is a template and not something we want to run,
We will disable the project, which is important
If the template contains build triggers
Or anything like that.

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a project named 'reindeer-template'. The browser address bar shows 'localhost:8080/view/Jenkins/job/reindeer-template/'. The page title is 'Project reindeer-template'. Below the title, it says 'Serves as a template for the Reindeer jobs.' and there is a link to 'edit description'. A green puzzle piece icon indicates 'ez-templates: This job is configured as a template'. A yellow warning triangle icon states 'This project is currently disabled' with an 'Enable' button next to it. Below this, there are links for 'Workspace' (with a folder icon) and 'Recent Changes' (with a notepad icon). The 'Permalinks' section is visible but empty. The footer shows 'Page generated: Oct 2, 2018 11:24:20 AM EDT', a link to 'REST API', and 'Jenkins ver. 2.143'.

reindeer-template [Jenkins] x +

localhost:8080/view/Jenkins/job/reindeer-template/

Jenkins > Jenkins > reindeer-template > [ENABLE AUTO REFRESH](#)

Project reindeer-template

Serves as a template for the Reindeer jobs.

[edit description](#)

ez-templates: This job is configured as a template

This project is currently disabled [Enable](#)

[Workspace](#)

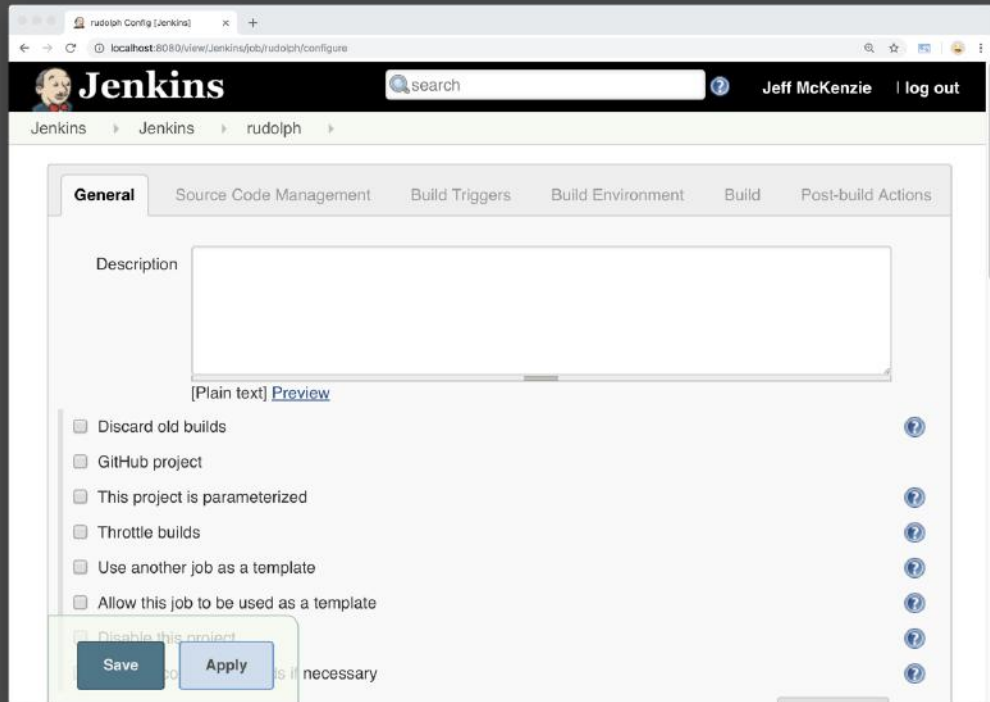
[Recent Changes](#)

Permalinks

Page generated: Oct 2, 2018 11:24:20 AM EDT [REST API](#) [Jenkins ver. 2.143](#)

So we will do that
And it shows up as disabled.

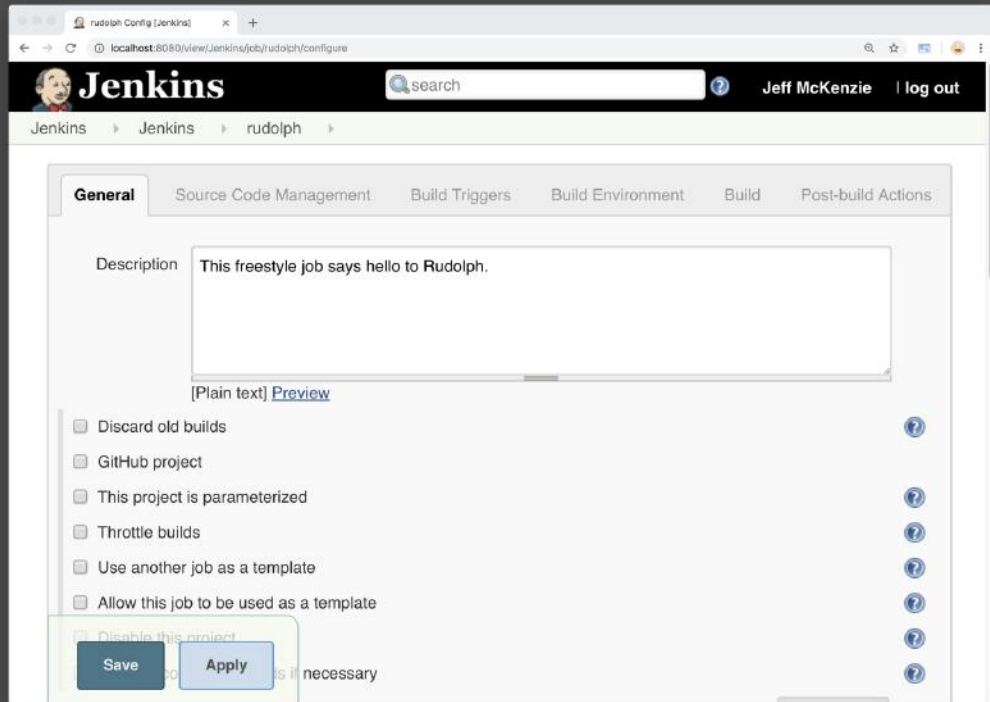
@jeffreymckenzie



Let's re-create our rudolph freestyle job
To see how to create a child job

We'll use the same description we had before....

@jeffreymckenzie



This freestyle job says hello to Rudolph....

Then we scroll down to the checkbox that says

“use another job as a template”

@jeffreymckenzie

The screenshot shows the Jenkins configuration page for a job named 'rudolph'. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/rudolph/configure`. The page has a breadcrumb trail: `Jenkins > Jenkins > rudolph`. The 'General' tab is selected, showing options for using another job as a template. The 'Name of template' dropdown is currently set to 'No template selected'. Below this, there are checkboxes for 'Allow this job to be used as a template', 'Disable this project', and 'Execute concurrent builds if necessary'. There are two 'Advanced...' buttons. The 'Source Code Management' section is partially visible at the bottom. A red tooltip box is overlaid on the 'Save' and 'Apply' buttons, containing the text: 'This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.' The tooltip also shows the 'Save' and 'Apply' buttons.

rudolph Config [Jenkins]

localhost:8080/view/Jenkins/job/rudolph/configure

Jenkins > Jenkins > rudolph

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

☒ Use another job as a template

Name of template No template selected

Advanced...

☐ Allow this job to be used as a template

☐ Disable this project

☐ Execute concurrent builds if necessary

Advanced...

Source Code Management

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.

Save Apply

When we check that we are given a dropdown
Of what template we want to select

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'rudolph'. The browser address bar indicates the URL is 'localhost:8080/view/Jenkins/job/rudolph/configure'. The 'General' tab is active, displaying several configuration options:

- ☒ Use another job as a template (with a help icon)
- Name of template:
- ☐ Allow this job to be used as a template (with a help icon)
- ☐ Disable this project (with a help icon)
- ☐ Execute concurrent builds if necessary (with a help icon)

Below these options are two 'Advanced...' buttons. The 'Source Code Management' section is partially visible at the bottom. A red warning box is overlaid on the 'Source Code Management' section, containing the following text:

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.

At the bottom of the warning box are two buttons: 'Save' and 'Apply'.

We will choose reindeer template

@jeffreymckenzie

The screenshot shows the Jenkins configuration page for a job named 'rudolph'. The 'General' tab is active. The 'Use another job as a template' checkbox is checked, and 'reindeer-template' is selected in the dropdown menu. Below this, a list of features is shown with checkboxes indicating which ones are retained from the template. A red warning box is overlaid on the 'Save' and 'Apply' buttons.

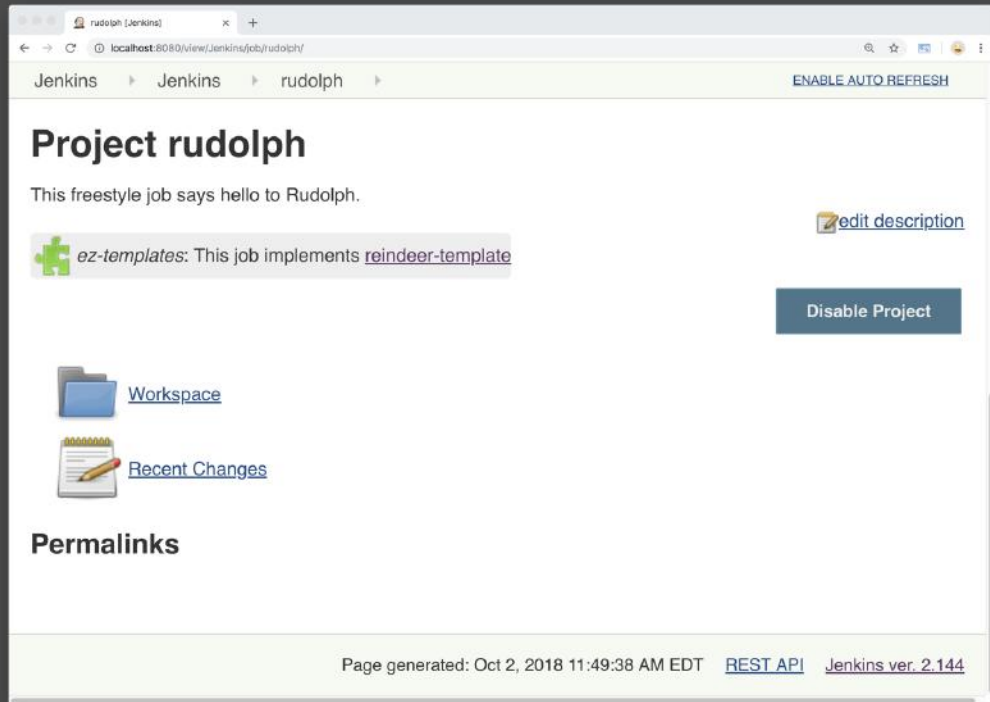
Feature	Retain
ez-templates	<input checked="" type="checkbox"/>
job-params	<input checked="" type="checkbox"/>
build-triggers	<input type="checkbox"/>
disabled	<input checked="" type="checkbox"/>
description	<input checked="" type="checkbox"/>

Warning: This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.

Buttons: Save, Apply

so Let's save that and go back to the project page

@jeffreymckenzie



This shows us that Rudolph
Is implementing the reindeer template.

Remember, all we did was add a description,
Select a template, and save it.

Let's go back into the configuration.

@jeffreymckenzie

The screenshot shows the Jenkins configuration page for a job named 'rudolph'. The browser address bar indicates the URL is 'localhost:8080/view/Jenkins/job/rudolph/configure'. The page has a breadcrumb trail: 'Jenkins > Jenkins > rudolph'. The 'General' tab is selected, showing a description field with the text 'This freestyle job says hello to Rudolph.' and a 'Discard old builds' section. The 'Discard old builds' section is checked, and the 'Strategy' is set to 'Log Rotation'. The 'Days to keep builds' field is empty, and the 'Max # of builds to keep' field is set to '3'. A red tooltip box is overlaid on the 'Save' and 'Apply' buttons, stating: 'This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.' The 'Save' button is blue, and the 'Apply' button is light blue. There is also an 'Advanced...' button at the bottom right.

rudolph Config [Jenkins]

localhost:8080/view/Jenkins/job/rudolph/configure

Jenkins > Jenkins > rudolph

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

Description This freestyle job says hello to Rudolph.

[Plain text] [Preview](#)

☒ Discard old builds

Strategy Log Rotation

Days to keep builds

Max # of builds to keep 3

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.

Save Apply

Advanced...

We have our description,
Our build history setting...

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'rudolph'. The 'General' tab is active, and the checkbox 'This project is parameterized' is checked. Below this, a list of 'String Parameter' entries is shown. The first parameter is 'GREETING', with a default value of 'Hello' and a description 'How to say Hello to the reindeer.' A red tooltip is overlaid on the 'Save' button, providing information about the template-based configuration. The second parameter, 'REINDEER', is partially visible below the first one.

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

☒ This project is parameterized

String Parameter

Name: GREETING

Default Value: Hello

Description: How to say Hello to the reindeer.

[Plain text] [Preview](#)

☐ Trim the string

String Parameter

Name: REINDEER

Default Value:

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are locked. Click Save to save the template.

Save Apply

Our GREETING parameter....

@jeffreymckenzie

The screenshot shows the Jenkins configuration page for a job named 'rudolph'. The 'String Parameter' section is visible, showing a parameter named 'REINDEER' with a default value and a description. A red tooltip box is overlaid on the 'Save' button.

String Parameter

Name: REINDEER

Default Value:

Description: The reindeer to say Hello to.

[Plain text] [Preview](#)

☐ Trim the string

Save **Apply**

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.

Our REINDEER parameter

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'rudolph'. The 'Source Code Management' tab is selected, showing options for 'None' and 'Git'. The 'Git' option is chosen. Under 'Repositories', a new repository is being added with the URL 'file:///Users/jmckenzie/projects/santa'. The 'Credentials' dropdown is set to '- none -' with an 'Add' button. Below this is an 'Advanced...' button and an 'Add Repository' button. The 'Branch Specifier (blank for \'any\')' is set to '*/master', with an 'Add Branch' button. A red warning box is overlaid on the bottom left, stating: 'This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are locked.' At the bottom of the configuration area are 'Save' and 'Apply' buttons.

rudolph Config [Jenkins]

localhost:8080/view/Jenkins/job/rudolph/configure

Jenkins > Jenkins > rudolph >

General **Source Code Management** Build Triggers Build Environment Build

Post-build Actions

Source Code Management

☐ None
☒ Git

Repositories

Repository URL

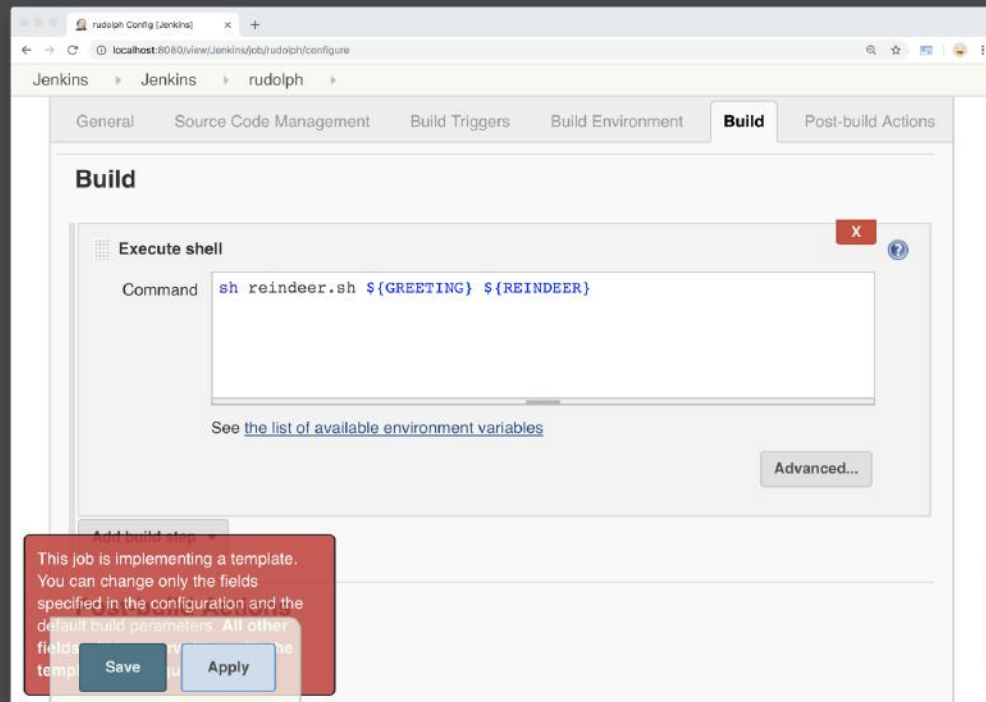
Credentials

Branch Specifier (blank for 'any')

This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are locked.

Our git settings...

@jeffreymckenzie



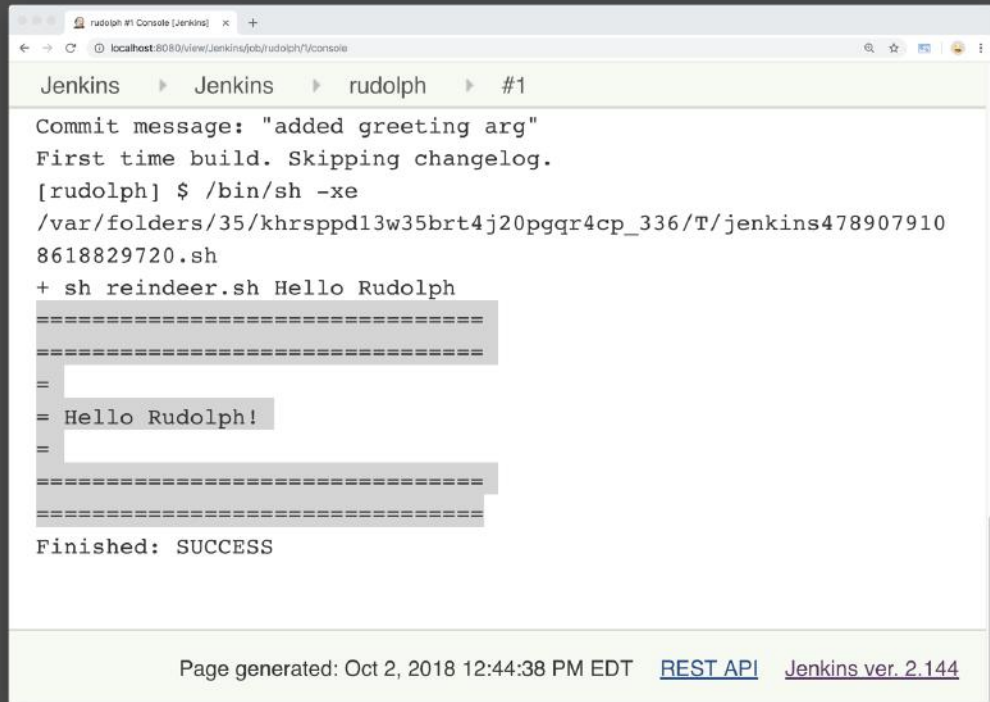
And our build step.

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a job named 'rudolph'. The 'General' tab is active, displaying a 'String Parameter' configuration. The parameter name is 'REINDEER', the default value is 'Rudolph', and the description is 'The reindeer to say Hello to.' There is a 'Save' button and an 'Apply' button. A red tooltip is visible over the 'Save' button, stating: 'This job is implementing a template. You can change only the fields specified in the configuration and the default build parameters. All other fields are inherited from the template.' The tooltip also includes a 'Save' button and an 'Apply' button. The configuration is currently based on the 'reindeer-template'.

So the only thing we have to do is
fill in the reindeer parameter
With Rudolph,
And run the project.

@jeffreymckenzie



The screenshot shows a web browser window displaying the Jenkins console for a job named 'rudolph'. The breadcrumb navigation at the top reads 'Jenkins > Jenkins > rudolph > #1'. The console output shows a commit message, a first-time build skipping changelog, and a shell command execution. The command 'sh reindeer.sh Hello Rudolph' is highlighted with a grey background. The output of the script is a series of lines, with 'Hello Rudolph!' being the visible result. The console ends with 'Finished: SUCCESS'. At the bottom, a footer indicates the page was generated on Oct 2, 2018, and provides links for the REST API and Jenkins version 2.144.

```
Jenkins > Jenkins > rudolph > #1
Commit message: "added greeting arg"
First time build. Skipping changelog.
[rudolph] $ /bin/sh -xe
/var/folders/35/khrsppd13w35brt4j20pgqr4cp_336/T/jenkins478907910
8618829720.sh
+ sh reindeer.sh Hello Rudolph
=====
=====
=
= Hello Rudolph!
=
=====
=====
Finished: SUCCESS

Page generated: Oct 2, 2018 12:44:38 PM EDT REST API Jenkins ver. 2.144
```

And our result is the same as before.

So now Santa can talk to as many reindeer as he wants,

And since they are all based on the same template,

The projects are much easier to change and maintain.

Those are the benefits, but there are limitations to this approach.

EZ Template Limitations

...**cannot override all settings**

...cannot be fully nested

...does not support pipeline jobs

First is not all settings can be overridden.

For example, if we wanted to modify

Our build step, our shell command, can only do so

In the template – local changes to that don't take effect

EZ Template Limitations

...cannot override all settings

...**cannot be fully nested**

...does not support pipeline jobs

Next, you can't really nest the templates effectively.

Would be nice to have a base template,

And have another template inherit from that.

But you can't, for example, add parameters to the child template,
Which would make it more useful.

EZ Template Limitations

...cannot override all settings

...cannot be fully nested

...**does not support pipeline jobs**

And lastly, templates only support freestyle jobs,
Not pipeline jobs.

Pipeline jobs is where the real power of Jenkins starts to come in.

So templates are a great option where you have a lot of freestyle jobs
That exhibit a similar pattern, and where parameters would be helpful.

@jeffreymckenzie

New Item [Jenkins]

localhost:8080/view/Jenkins/newJob

Jenkins

search? Jeff McKenzie | log out

Enter an item name

rudolph-pipeline

» 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

☒ Add to current view

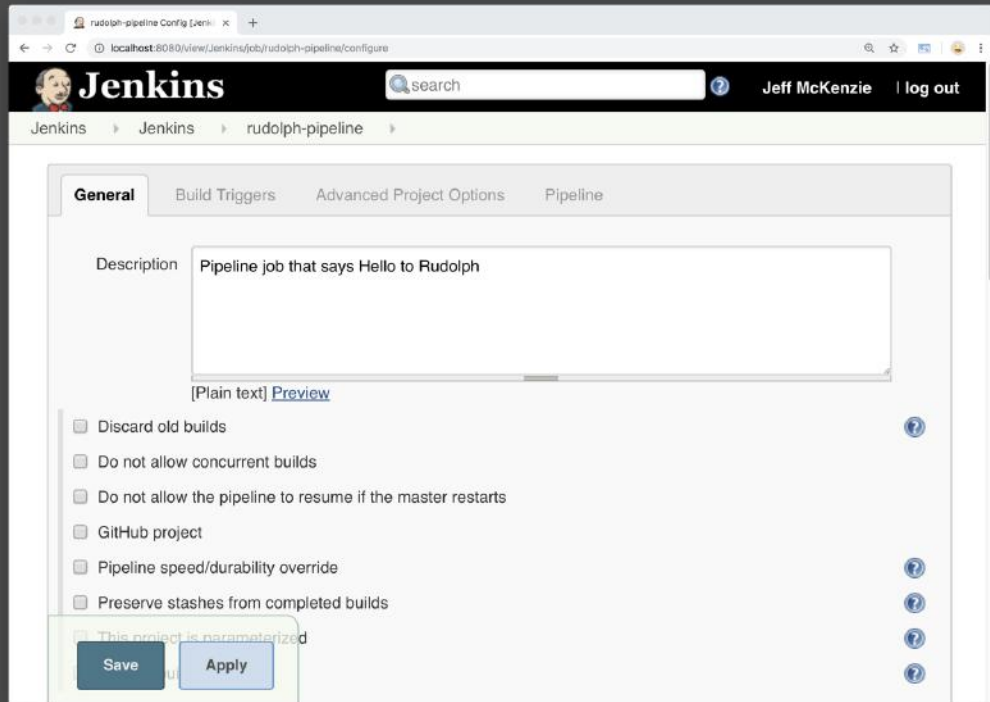
Folder

So let's turn to our next option for scaling,
The pipeline job.

Let's create one called Rudolph pipeline
that does the same thing
Our freestyle job does, to show the differences.

-- comment on pipeline description

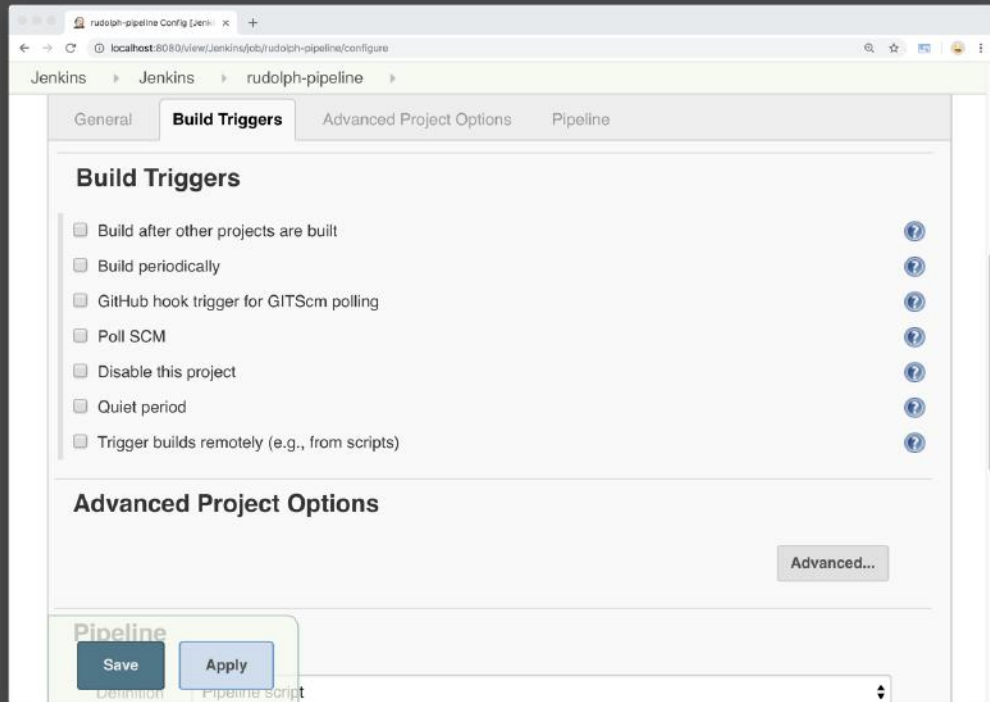
@jeffreymckenzie



We will add our description,
And notice that this has less configuration options
Than a freestyle build.

We have the General section...

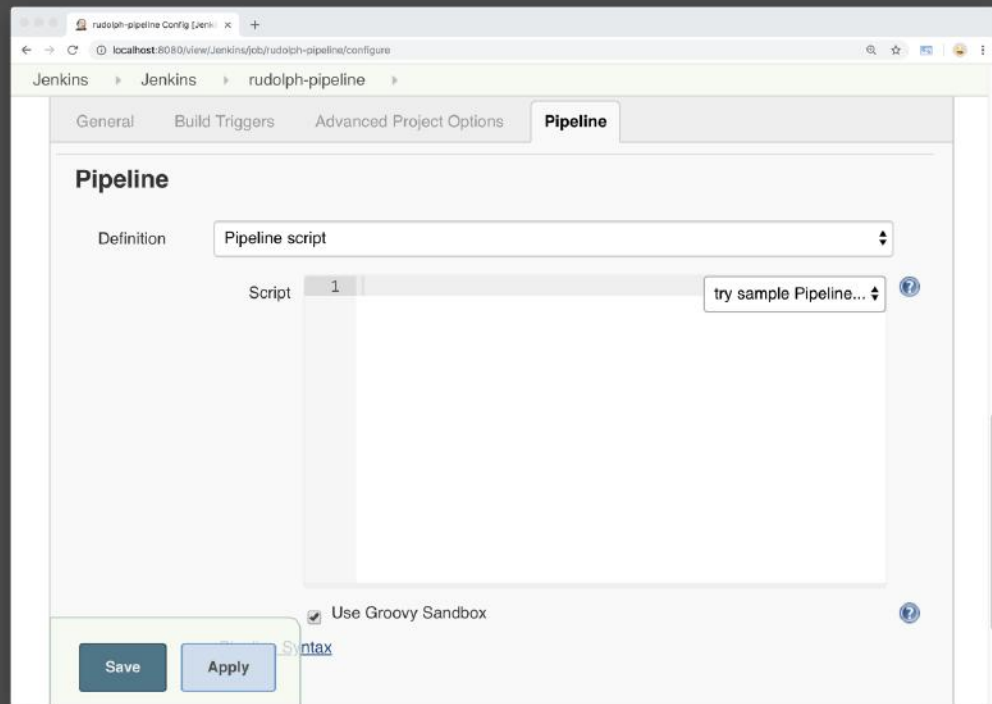
@jeffreymckenzie



The build triggers section,
And the advanced project options.

But the main part of the pipeline job is the script

@jeffreymckenzie



Let's take a close-up look at that script

@jeffreymckenzie

```
pipeline {
  agent any
  options {
    buildDiscarder(logRotator(numToKeepStr:'3'))
  }
  parameters {
    string(
      name: 'GREETING',
      defaultValue: 'Hello',
      description: 'The way to say hello.'
    )
    string(
      name: 'REINDEER',
      defaultValue: 'Rudolph',
      description: 'The reindeer to say hello to.'
    )
  }
  stages {
    stage('Say Hi to Rudolph') {
      steps {
        checkout scm: [
          $class: 'GitSCM',
          branches: [
            [name: '*/master']
          ],
          userRemoteConfigs: [
            [url: 'file:///Users/jmckenzie/projects/santa']
          ]
        ]
        sh 'sh reindeer.sh "${GREETING}" "${REINDEER}"'
      }
    }
  }
}
```

Here's the whole thing – As you can see, not a lot of code here

let's break this down

```
pipeline {  
  agent any  
  options {  
    buildDiscarder(logRotator(numToKeepStr:'3'))  
  }  
}
```

On the Jenkins site, there's a reference area
for all of these commands and syntax,
So you can look all this stuff up

A pipeline script has to start with the pipeline block.
Then you specify the agent, or where this is going to run

For our purposes this can run anywhere...

We can set the build history to keep the last 3 builds,
just like we did in the freestyle job


```
pipeline {  
  agent any  
  options {  
    buildDiscarder(logRotator(numToKeepStr:'3'))  
  }  
  parameters {  
    string( name: 'GREETING',  
            defaultValue: 'Hello',  
            description: 'The way to say hello. ')  
  }  
}
```

Then we can add the greeting parameter,
Again, same settings and values

```
pipeline {  
  agent any  
  options { ... }  
  parameters {  
    ...  
    string( name: 'REINDEER',  
            defaultValue: 'Rudolph',  
            description: 'The reindeer to say hello to.')  
  }  
}
```

Add our reindeer parameter...

```
pipeline {  
  agent any  
  options { ... }  
  parameters { ... }  
  stages {  
    stage('Say Hi to Rudolph') {  
      steps {  
      }  
    }  
  }  
}
```

Then we add the stages block,
Which is just a way to group commands into sections.

We have only one stage, and we can give it a name...
And within a stage, we have steps.

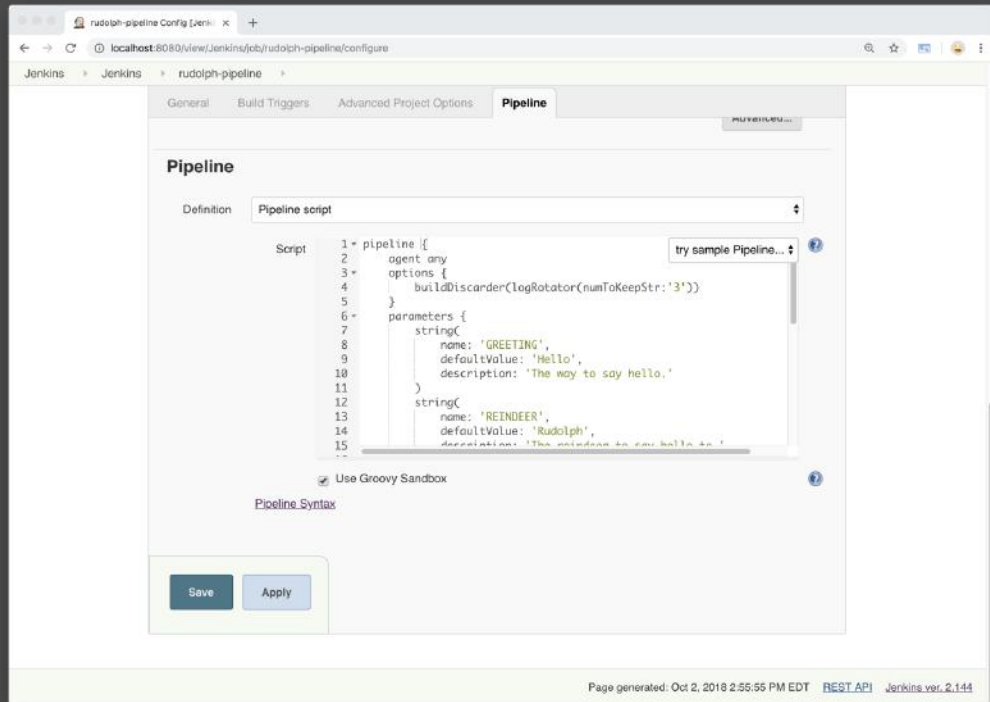
```
pipeline {  
  ...  
  stage('Say Hi to Rudolph') {  
    steps {  
      checkout scm: [$class: 'GitSCM',  
        branches: [[name: '*/master']],  
        userRemoteConfigs: [[url:  
          'file:///Users/jmckenzie/projects/santa']]  
    }  
  }  
}
```

Our first step will be do to the checkout
From the git repository

```
pipeline {  
  ...  
  stage('Say Hi to Rudolph') {  
    steps {  
      checkout scm:[ ... ]  
      sh 'sh reindeer.sh  
        "${GREETING}" "${REINDEER}"'  
    }  
  }  
}
```

And the second step is to execute the shell script
Using the SH command and our parameters.

@jeffreymckenzie



So here's what it looks like back in the pipeline project...

And let's run that and see what we get

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a pipeline named 'rudolph-pipeline'. The left sidebar contains navigation links: Back to Dashboard, Status, Changes, Build Now, Delete Pipeline, Configure, Full Stage View, Rename, and Pipeline Syntax. The main content area is titled 'Pipeline rudolph-pipeline' and includes a description 'Pipeline job that says Hello to Rudolph.' and buttons for 'edit description' and 'Disable Project'. Below this is a 'Recent Changes' section. The 'Stage View' section displays a bar chart of stage times. The 'Build History' section shows a table of recent builds. The 'Permalinks' section is at the bottom.

Pipeline rudolph-pipeline
Pipeline job that says Hello to Rudolph.

[edit description](#)
[Disable Project](#)

[Recent Changes](#)

Stage View

Average stage times:
(Average full run time: ~1s)

Stage	Time
Say Hi to Rudolph	584ms
584ms	584ms

Build History [trend](#)

Build	Time
1	Oct 2, 2018 2:59 PM

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

Permalinks

Page generated: Oct 2, 2018 2:59:08 PM EDT [REST API](#) Jenkins ver. 2.144

We have a successful build, and if you'll notice
There's a stage view here as well....

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a pipeline named 'rudolph-pipeline'. The breadcrumb navigation at the top reads 'Jenkins > Jenkins > rudolph-pipeline'. The main heading is 'Pipeline rudolph-pipeline', followed by the description 'Pipeline job that says Hello to Rudolph.' On the right, there are links for 'edit description' and a 'Disable Project' button. Below the description is a 'Recent Changes' link with a notepad icon. The 'Stage View' section displays a table of stage execution data.

Average stage times: (Average full run time: ~1s)		Say Hi to Rudolph
		584ms
#1	Oct 02 14:59 No Changes	584ms

Which shows the name of the stage
And how long it took to run.

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a pipeline named 'rudolph-pipeline'. The 'Pipeline' tab is selected, and the 'Definition' dropdown is set to 'Pipeline script'. A text area contains a Groovy script defining a pipeline with two parameters: 'GREETING' (defaulting to 'Hello') and 'REINDEER' (defaulting to 'Rudolph'). The 'Use Groovy Sandbox' checkbox is checked. At the bottom, there are 'Save' and 'Apply' buttons. A footer at the bottom right indicates the page was generated on Oct 2, 2018, and is for Jenkins version 2.144.

```
1 pipeline {
2   agent any
3   options {
4     buildDiscarder(logRotator(numToKeepStr: '3'))
5   }
6   parameters {
7     string(
8       name: 'GREETING',
9       defaultValue: 'Hello',
10      description: 'The way to say hello,'
11    )
12    string(
13      name: 'REINDEER',
14      defaultValue: 'Rudolph',
15      description: 'The reindeer to say hello to'
16    )
17  }
```

☒ Use Groovy Sandbox

Save Apply

Page generated: Oct 2, 2018 2:55:55 PM EDT [REST API](#) Jenkins ver. 2.144

If we go back to our pipeline,
There's another option here for definition...

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a pipeline. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/rudolph-pipeline/configure`. The page has a breadcrumb trail: `Jenkins > Jenkins > rudolph-pipeline`. There are four tabs: `General`, `Build Triggers`, `Advanced Project Options`, and `Pipeline` (which is active). The `Pipeline` tab contains the following configuration options:

- Definition:** A dropdown menu set to `Pipeline script from SCM`.
- SCM:** A dropdown menu set to `None`.
- Script Path:** A text input field containing `Jenkinsfile`.
- Lightweight checkout:** A checkbox that is checked.

Below these fields is a link labeled `Pipeline Syntax`. At the bottom left of the configuration area are two buttons: `Save` and `Apply`. The footer of the page reads: `Documentation: Oct 3, 2016 2:04:16 PM EDT - REST API - Jenkins v2.14.4`

Which is “pipeline script from SCM”
Or source control management,

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a pipeline. The browser address bar indicates the URL is `localhost:8080/view/Jenkins/job/rudolph-pipeline/configure`. The page has tabs for 'General', 'Build Triggers', 'Advanced Project Options', and 'Pipeline', with 'Pipeline' being the active tab. The 'Pipeline' section is titled 'Pipeline' and contains the following fields:

- Definition:** A dropdown menu set to 'Pipeline script from SCM'.
- SCM:** A dropdown menu set to 'Git'.
- Repositories:** A section containing:
 - Repository URL:** A text field with the value `file:///Users/mckenzie/projects/aria`.
 - Credentials:** A dropdown menu set to 'none' with an 'Add' button.
 - Advanced...** and **Add Repository** buttons.
- Branches to build:** A section containing:
 - Branch Specifier (blank for 'any'):** A text field with the value `*/master`.
 - Add Branch** button.
- Repository browser:** A dropdown menu set to '(Auto)'.
- Additional Behaviours:** A dropdown menu set to 'Add'.
- Script Path:** A text field with the value `Jenkinsfile`.
- Lightweight checkout:** A checkbox that is checked.
- Pipeline Syntax:** A link.

At the bottom of the configuration area are two buttons: 'Save' and 'Apply'.

We can choose a git repo where this is coming from,
Which is great because it allows us
To version control that pipeline script
Just like we would with any other code.

@jeffreymckenzie

```
pipeline {
  agent any
  options {
    buildDiscarder(logRotator(numToKeepStr:'3'))
  }
  parameters {
    string(
      name: 'GREETING',
      defaultValue: 'Hello',
      description: 'The way to say hello.'
    )
    string(
      name: 'REINDEER',
      defaultValue: 'Rudolph',
      description: 'The reindeer to say hello to.'
    )
  }
  stages {
    stage('Say Hi to Rudolph') {
      steps {
        checkout scm: [
          $class: 'GitSCM',
          branches: [
            [name: '*/master']
          ],
          userRemoteConfigs: [
            [url: 'file:///Users/jmckenzie/projects/santa']
          ]
        ]
        sh 'sh reindeer.sh "${GREETING}" "${REINDEER}"'
      }
    }
  }
}
```

So that's great and everything,
But Santa has a problem –
wants to communicate with all his reindeer,

now only rudolph...

@jeffreymckenzie



He has to copy this script
to any pipeline project that needs it.

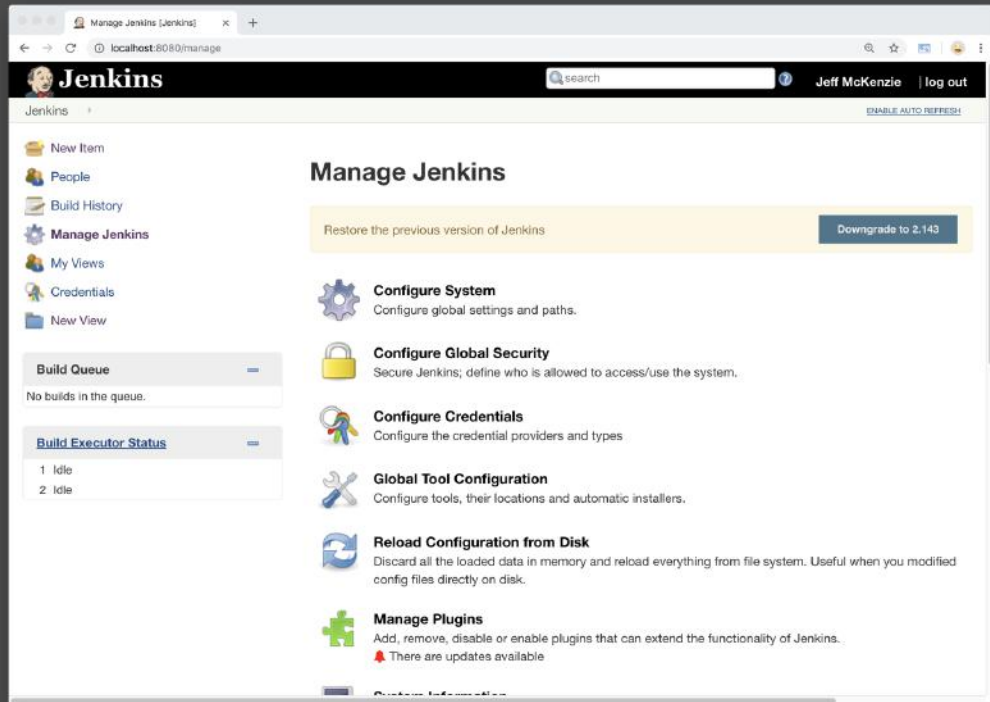
So let's fix that.

=====

[https://commons.wikimedia.org/wiki/File:Mr_Santa_Claus_\(HS85-10-30308\).jpg](https://commons.wikimedia.org/wiki/File:Mr_Santa_Claus_(HS85-10-30308).jpg)

British Library [Public domain or Public domain], via Wikimedia
Commons

@jeffreymckenzie



If we go to the manage Jenkins page,
Into Configure System...

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface in a web browser. The browser's address bar displays 'localhost:8080/configure'. The Jenkins header includes the logo, a search bar, the user name 'Jeff McKenzie', and a 'log out' link. The left sidebar contains navigation links: 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Credentials', and 'New View'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two idle executors). The main configuration area is titled 'configuration' and contains several sections: 'Home directory' (set to '/Users/jmckenzie/.jenkins'), 'System Message' (a large text area), '# of executors' (set to '2'), 'Labels' (empty), 'Usage' (set to 'Use this node as much as possible'), 'Quiet period' (set to '5'), 'SCM checkout retry count' (set to '0'), 'Default view' (set to 'all'), 'Restrict project naming' (unchecked), 'Global properties' (with 'Environment variables' and 'Tool Locations' unchecked), and 'Pipeline Speed/Durability Settings' (with 'Pipeline Default Speed/Durability Level' set to 'None: use pipeline default (MAX_SURVIVABILITY)'). At the bottom are 'Save' and 'Apply' buttons.

And we scroll down a bit,
We get to a section called Global Pipeline Libraries

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface in a web browser. The browser's address bar displays 'localhost:8080/configure'. The page has a breadcrumb trail 'Jenkins > configuration'. The main content area is titled 'Global Pipeline Libraries' and includes a description: 'Sharable libraries available to any Pipeline jobs running on this system. These libraries will be trusted, meaning they run without "sandbox" restrictions and may use @Grab.' Below this is an 'Add' button. The next section is 'Build-timeout Plugin > BuildStep Action', which contains a checkbox labeled 'Enable BuildStep Action'. The 'Git plugin' section follows, featuring two input fields: 'Global Config user.name Value' and 'Global Config user.email Value'. At the bottom of the configuration area are 'Save' and 'Apply' buttons.

Which says...

Shareable libraries available to any Pipeline jobs

Running on this system.

That means we can create shared code
(written in groovy)

Accessible to any pipeline script.

To do that, we click add...

@jeffreymckenzie

The screenshot shows the Jenkins configuration page for a Library. The browser address bar indicates the URL is localhost:8080/configure. The page title is "Jenkins configuration". The "Library" section is active, showing fields for "Name" and "Default version", both of which are empty. A red error message "You must enter a name." is displayed next to the "Name" field. Below these fields are three checkboxes: "Load implicitly" (unchecked), "Allow default version to be overridden" (checked), and "Include @Library changes in job recent changes" (checked). The "Retrieval method" section shows "Modern SCM" selected. The "Source Code Management" section is partially visible at the bottom. At the bottom of the page are "Save" and "Apply" buttons.

Configure System [Jenkins] x +

localhost:8080/configure

Jenkins configuration

Library

Name ?

You must enter a name.

Default version ?

Load implicitly ☐ ?

Allow default version to be overridden ☒ ?

Include @Library changes in job recent changes ☒ ?

Retrieval method

☒ Modern SCM ?

Source Code Management

Save Apply

Then you'll add information about your library
-- a name: we'll call it Jenkins-lib

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface for a Library. The browser address bar indicates the URL is localhost:8080/configure. The page title is "Jenkins configuration".

Library

Name: ⓘ

Default version: ⓘ

Load implicitly: ☐ ⓘ

Allow default version to be overridden: ☒ ⓘ

Include @Library changes in job recent changes: ☒ ⓘ

Retrieval method

☒ Modern SCM ⓘ

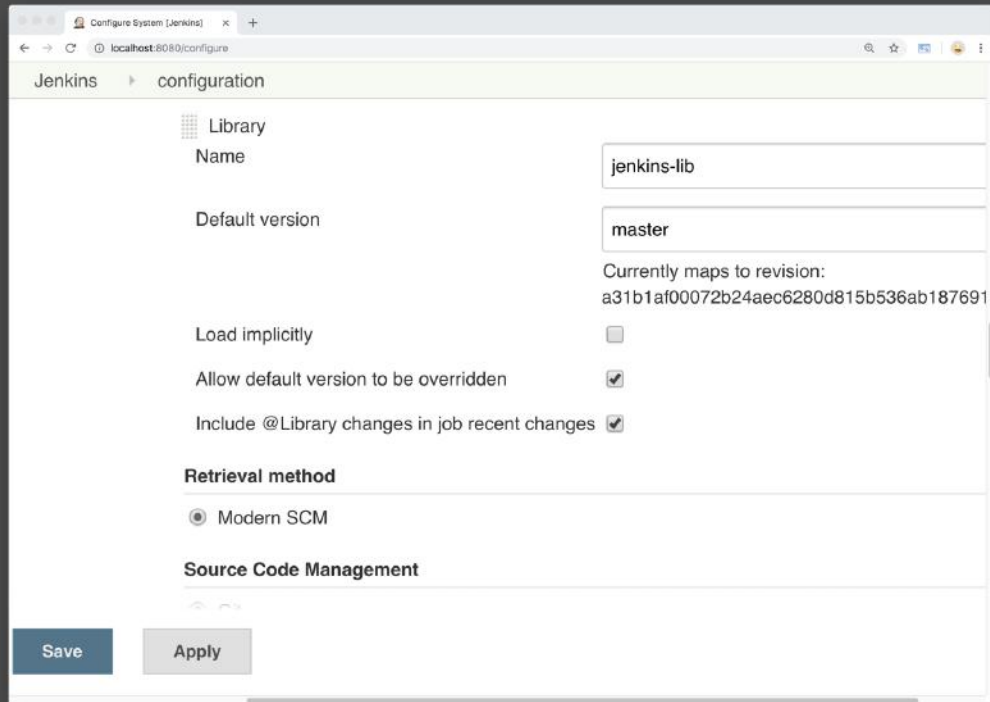
Source Code Management

☒ Git

Project Repository: ⓘ

At the bottom, there are two buttons: "Save" and "Apply".

@jeffreymckenzie



The screenshot shows the Jenkins configuration page for a Library. The browser address bar indicates the URL is localhost:8080/configure. The page title is "Jenkins configuration". The "Library" section is expanded, showing the following fields:

- Name:** jenkins-lib
- Default version:** master
- Currently maps to revision:** a31b1af00072b24aec6280d815b536ab187691
- Load implicitly:** ☐
- Allow default version to be overridden:** ☒
- Include @Library changes in job recent changes:** ☒

The **Retrieval method** section shows **Modern SCM** selected with a radio button. Below this is the **Source Code Management** section, which is currently collapsed. At the bottom of the form are two buttons: **Save** and **Apply**.

Default version refers to version control,
So we will just list the master branch

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface in a web browser. The browser's address bar displays 'localhost:8080/configure'. The page title is 'Jenkins configuration'. The main content area is titled 'Source Code Management' and includes the following sections:

- Retrieval method:** A radio button is selected for 'Modern SCM'.
- Source Code Management:**
 - Git:** A radio button is selected.
 - Project Repository:** An empty text input field.
 - Credentials:** A dropdown menu showing '- none -' and an 'Add' button with a key icon.
 - Behaviors:** A section with two checkboxes: 'Within Repository' and 'Discover branches', both of which are currently unchecked.

At the bottom of the configuration area, there is a red 'Delete' button. At the bottom of the page, there are two buttons: 'Save' and 'Apply'.

Then we add the git repo
we want to pull the shared code from

@jeffreymckenzie

The screenshot shows the Jenkins configuration interface in a web browser. The browser's address bar shows 'localhost:8080/configure'. The page title is 'Jenkins configuration'. The 'Retrieval method' section has 'Modern SCM' selected. The 'Source Code Management' section has 'Git' selected. The 'Project Repository' field contains the local file path 'file:///Users/jmckenzie/projects/jenkins-lib'. The 'Credentials' dropdown is set to '- none -' with an 'Add' button next to it. The 'Behaviors' section includes 'Within Repository' and 'Discover branches' (which has a 'Delete' button next to it). At the bottom, there are 'Save' and 'Apply' buttons.

Configure System [Jenkins] x +

localhost:8080/configure

Jenkins configuration

Retrieval method

☒ Modern SCM

Source Code Management

☒ Git

Project Repository

Credentials

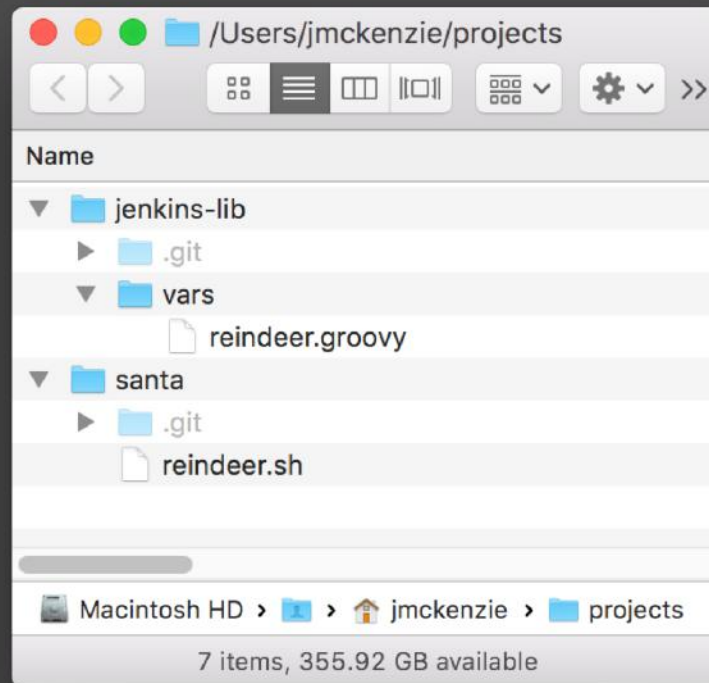
Behaviors

Within Repository

Discover branches

Again we'll point that to a separate git repo locally
And click save

@jeffreymckenzie



So these are our 2 git repos –
Santa, which contains our shell script,

And Jenkins-lib, which contains
Our global pipeline library

Jenkins expects to see a directory
Called VARS, with groovy scripts
underneath that

-- let's look at the reindeer groovy script
in the library

```
- reindeer.groovy
#!/usr/bin/env groovy
def call(Map<String, Object> options) {
    try {
        node {
            String greeting = options."greeting".toString()
            String reindeer = options."reindeer".toString()
            stage("Say Hi to ${reindeer}") {
                checkout scm: [
                    $class: 'GitSCM',
                    branches: [[name: '*/master']],
                    userRemoteConfigs: [[url:
                        'file:///Users/jmckenzie/projects/santa']]
                sh "sh reindeer.sh ${greeting} ${reindeer}"
            }
        }
    }
    catch (Throwable err) {
        throw err
    }
}
```

so this is a rewrite of our pipeline script in groovy
Groovy is a scripting language
Similar to ruby,
that runs on the JVM

This is even less code than our pipeline script.
Let's look at it in more detail.

```
- reindeer.groovy

#!/usr/bin/env groovy
def call(Map<String, Object> options) {
    try {
        node {

        }
    }
    catch (Throwable err) {
        throw err
    }
}
```

You create a method using the DEF keyword,
And then name your method “call” --
You refer to the method in Jenkins by the file name
Rather than the actual method name,
Which we will see in a little bit.

We’re going to pass in a Map object called “options”
Which is simply a key-value pair collection.
We can use try and catch,
And the node block in where we place
What will run in the pipeline


```
- reindeer.groovy

#!/usr/bin/env groovy
def call(Map<String, Object> options) {
    try {
        node {
            String greeting =
                options."greeting".toString()
            String reindeer =
                options."reindeer".toString()
        }
    }
    catch (Throwable err) { ... }
}
```

Then we declare two variables –
Greeting and reindeer, that hold our parameters.

```
- reindeer.groovy

def call(Map<String, Object> options) {
    ...
    String greeting = ...
    String reindeer = ...
    stage("Say Hi to ${reindeer}") {
        }
    }
    }
    catch (Throwable err) { ... }
}
```

Then we add the stage block –
And here we can actually incorporate
The variable name into the Jenkins output

```
- reindeer.groovy

def call(Map<String, Object> options) {
    ...
    stage("Say Hi to ${reindeer}") {
        checkout scm: [
            $class: 'GitSCM',
            branches: [[name: '*/master']],
            userRemoteConfigs: [[url: 'file:
            ///Users/jmckenzie/projects/santa']]
        }
    }
    ...
}
```

Then the git checkout from our santa repo,
to pull our reindeer shell script

```
- reindeer.groovy

def call(Map<String, Object> options) {
    ...
    stage("Say Hi to ${reindeer}") {
        checkout scm: [ ... ]
        sh "sh reindeer.sh ${greeting}
            ${reindeer}"
    }
    ...
}
```

And finally our call to the shell script,
Where we can pass the variables in.

Let's create a new pipeline job and see how this works.

@jeffreymckenzie

New Item [Jenkins]

localhost:8080/view/all/newJob


Jenkins

search ? Jeff McKenzie | log out


Jenkins > All >

Enter an item name


» 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

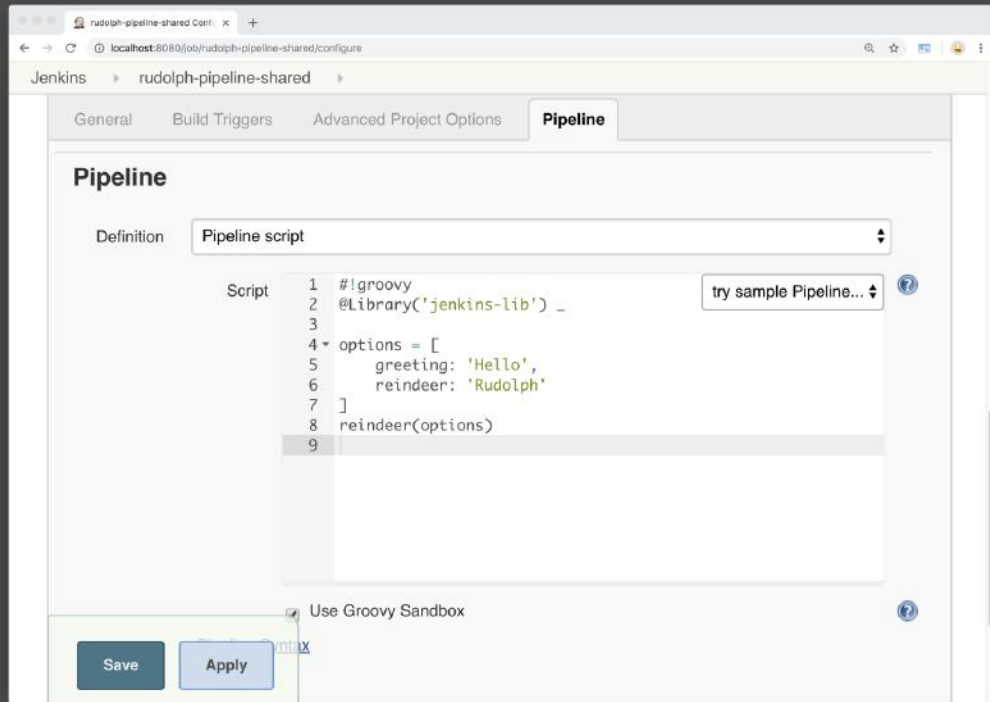
We'll call it Rudolph-pipeline-shared

@jeffreymckenzie

The screenshot shows the Jenkins web interface. At the top, the Jenkins logo is on the left, and the user 'Jeff McKenzie' with a 'log out' link is on the right. Below the header, a breadcrumb trail shows 'Jenkins' > 'rudolph-pipeline-shared'. The main content area has four tabs: 'General' (selected), 'Build Triggers', 'Advanced Project Options', and 'Pipeline'. In the 'General' tab, there is a 'Description' field containing the text 'Say hello to Rudolph with a shared groovy method.' Below the text area are links for '[Plain text]' and 'Preview'. At the bottom of the configuration area, there are two buttons: 'Save' and 'Apply'. Below these buttons, there are two checkboxes: 'Discard old builds' and 'Do not allow the pipeline to resume if the master restarts'. A help icon is visible on the right side of the configuration area.

We're going to say hello to Rudolph with a shared groovy method.

@jeffreymckenzie



We go down to the pipeline script area
And just add a few lines here –

And that's it –

Lets take a closer look.

```
#!groovy
@Library('jenkins-lib') _

options = [
    greeting: 'Hello',
    reindeer: 'Rudolph'
]
reindeer(options)
```

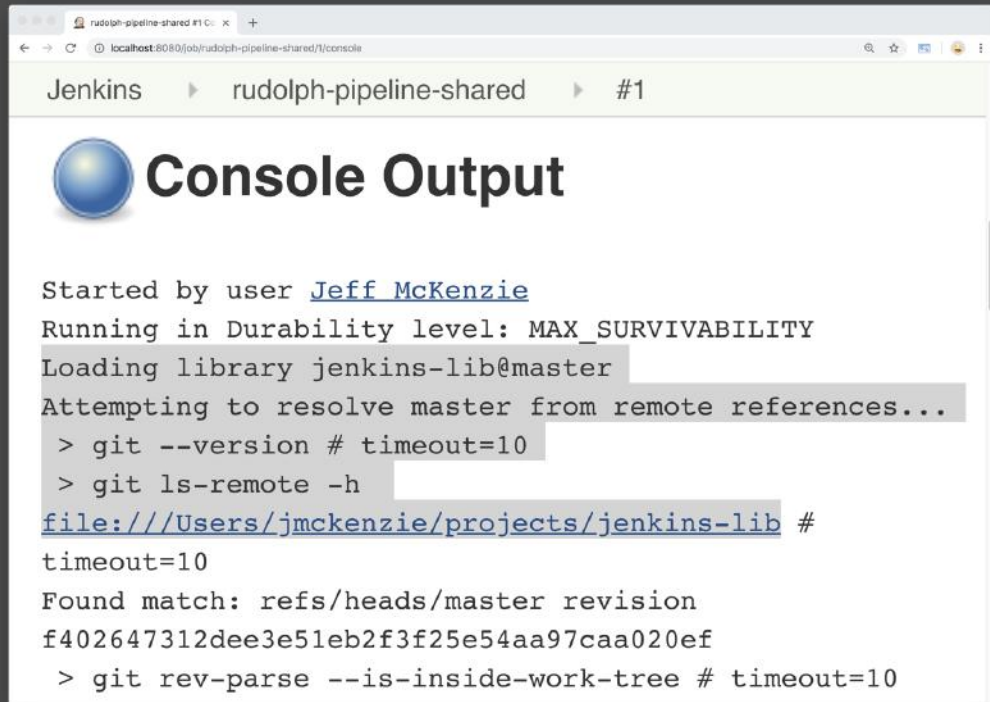
The first line tells Jenkins you're executing groovy –
Second line is a reference to the global library
You want to use – (note underscore)

Then you create the map object with the variable values,

And call the method by passing in the options.
Jenkins will look for the file called reindeer
and execute the method in it

Let's run this and look at the result...

@jeffreymckenzie

A screenshot of a web browser displaying the Jenkins console output for a job named 'rudolph-pipeline-shared'. The browser's address bar shows 'localhost:8080/job/rudolph-pipeline-shared/1/console'. The Jenkins breadcrumb navigation shows 'Jenkins > rudolph-pipeline-shared > #1'. The main heading is 'Console Output' with a blue sphere icon. The output text shows the job was started by 'Jeff McKenzie', running in 'MAX_SURVIVABILITY' durability mode, and loading the 'jenkins-lib@master' library. It then attempts to resolve the master from remote references using a series of git commands: 'git --version', 'git ls-remote -h', and 'git rev-parse --is-inside-work-tree', all with a 10-second timeout. The output shows a match found for the master branch at revision 'f402647312dee3e51eb2f3f25e54aa97caa020ef'.

First you can see Jenkins is loading the library from git...

@jeffreymckenzie

```
Jenkins ▶ rudolph-pipeline-shared ▶ #1

[Pipeline] checkout
  > git rev-parse --is-inside-work-tree # timeout=10
  Fetching changes from the remote Git repository
  > git config remote.origin.url
  file:///Users/jmckenzie/projects/santa # timeout=10
  Fetching upstream changes from
  file:///Users/jmckenzie/projects/santa
  > git --version # timeout=10
  > git fetch --tags --progress
  file:///Users/jmckenzie/projects/santa
  +refs/heads/*:refs/remotes/origin/*
  > git rev-parse refs/remotes/origin/master^{commit} #
  timeout=10
  > git rev-parse
  refs/remotes/origin/origin/master^{commit} #
```

Then it gets changes from the Santa repo...

@jeffreymckenzie



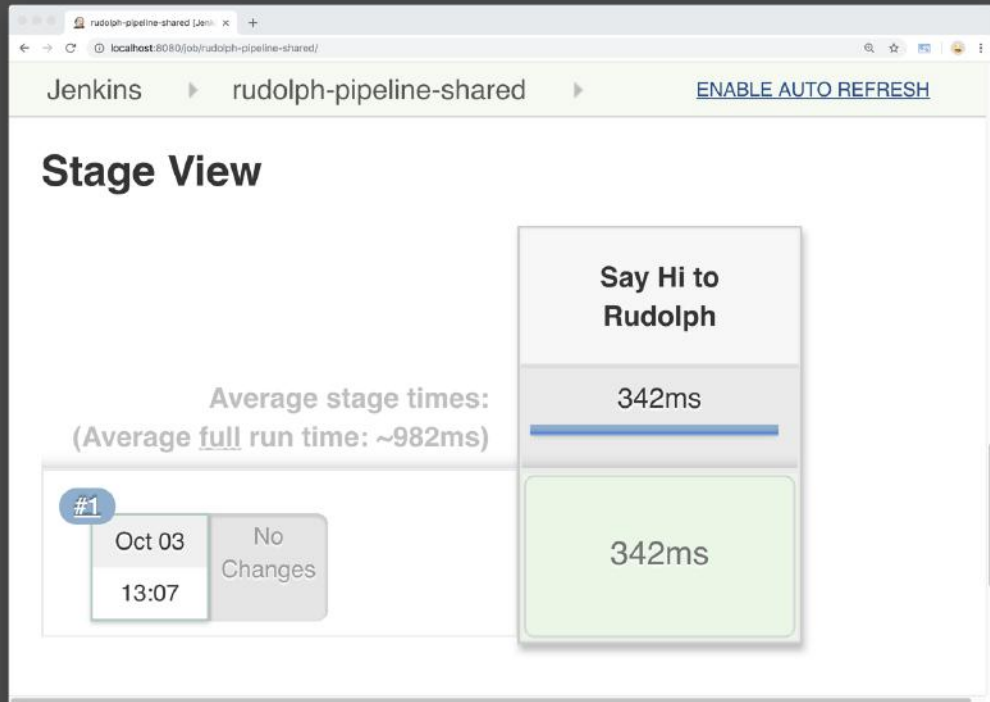
The screenshot shows a web browser window displaying the Jenkins console for a pipeline named 'rudolph-pipeline-shared' at build #1. The breadcrumb navigation at the top reads 'Jenkins > rudolph-pipeline-shared > #1'. The console output shows the pipeline starting with '[Pipeline] sh', followed by '[rudolph-pipeline-shared] Running shell script'. A command is executed: '+ sh reindeer.sh Hello Rudolph'. The output of this command is a series of equals signs followed by 'Hello Rudolph!'. The pipeline then continues with '[Pipeline] }', '[Pipeline] // stage', '[Pipeline] }', '[Pipeline] // node', and finally '[Pipeline] End of Pipeline'.

```
[Pipeline] sh
[rudolph-pipeline-shared] Running shell script
+ sh reindeer.sh Hello Rudolph
=====
=====
=
= Hello Rudolph!
=
=====
=====

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
```

Then it runs our script
and we get the expected output...

@jeffreymckenzie



And when we look at the stage view
We can see our variable value (Rudolph)
Was used to name it.

@jeffreymckenzie



So that's better – we have a very small project/job
That uses shared code and is easy to maintain.

However, we still have to create all those jobs.

Now Santa's a busy man...

=====

<https://commons.wikimedia.org/wiki/File:HanRenne.JPG>

By GrottesdeHan [CC BY-SA 3.0

(<https://creativecommons.org/licenses/by-sa/3.0/>), from Wikimedia
Commons

@jeffreymckenzie



He and Mrs claus have a lot to do around Christmas,
And he really doesn't want to have create all those individual jobs.

He's already automated his communication with his reindeer –

Wouldn't it be nice if he could automate his automation?

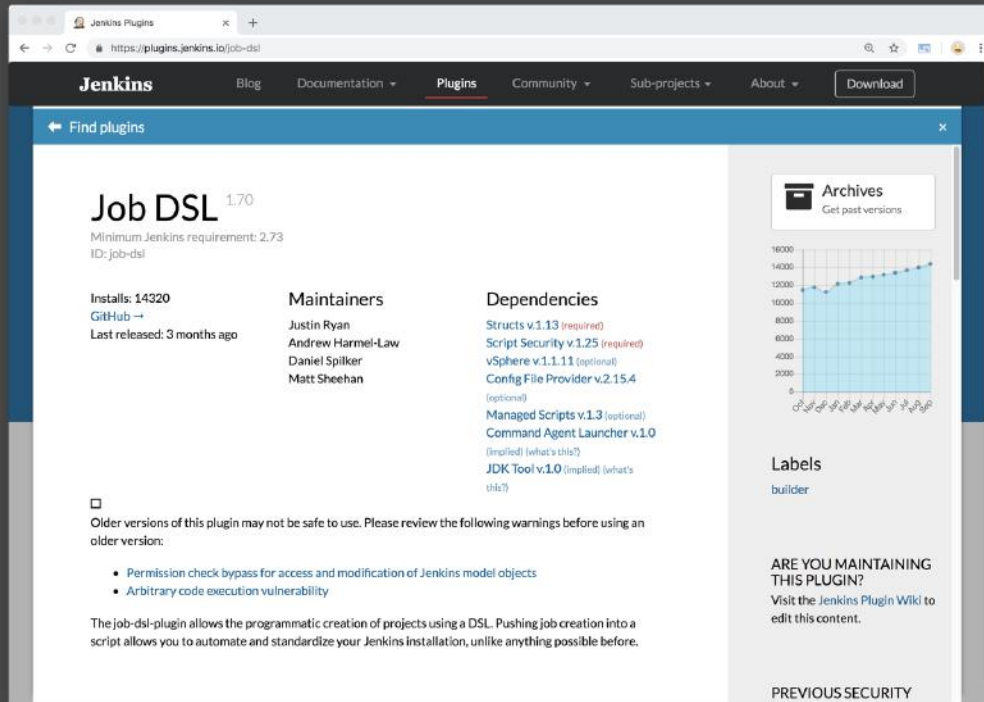
Well it turns out he can...

=====

https://commons.wikimedia.org/wiki/File:Christmas_came_early_for_one_Alaska_village_151016-Z-MW427-433.jpg

By Staff Sgt. Edward Eagerton [Public domain], via Wikimedia Commons

@jeffreymckenzie



The screenshot shows the Jenkins Job DSL plugin page. The page header includes the Jenkins logo and navigation links: Blog, Documentation, Plugins (active), Community, Sub-projects, and About. A 'Download' button is also present. The main content area is titled 'Find plugins' and displays the 'Job DSL 1.70' plugin. It includes the following information:

- Installs:** 14320
- GitHub:** →
- Last released:** 3 months ago
- Maintainers:** Justin Ryan, Andrew Harmel-Law, Daniel Spilker, Matt Sheehan
- Dependencies:** Structs v.1.13 (required), Script Security v.1.25 (required), vSphere v.1.1.11 (optional), Config File Provider v.2.15.4 (optional), Managed Scripts v.1.3 (optional), Command Agent Launcher v.1.0 (modified) (what's this?), JDK Tool v.1.0 (modified) (what's this?)
- Archives:** Get past versions
- Labels:** builder
- ARE YOU MAINTAINING THIS PLUGIN?** Visit the Jenkins Plugin Wiki to edit this content.
- PREVIOUS SECURITY** (with a list of security advisories)

A warning box states: 'Older versions of this plugin may not be safe to use. Please review the following warnings before using an older version:'

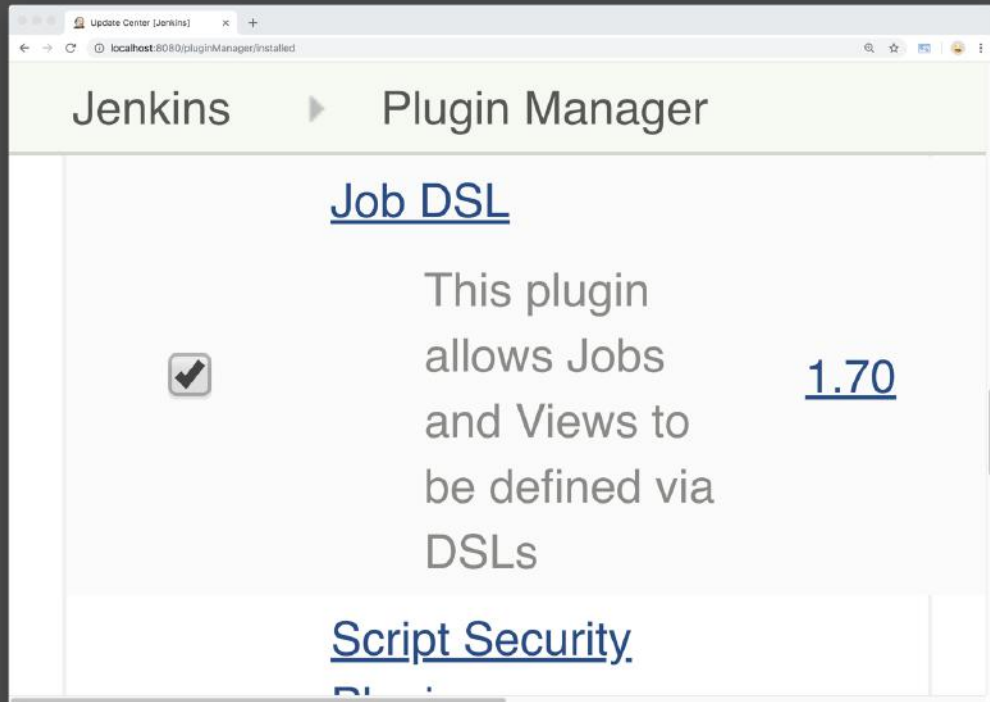
- Permission check bypass for access and modification of Jenkins model objects
- Arbitrary code execution vulnerability

The bottom of the page mentions: 'The job-dsl-plugin allows the programmatic creation of projects using a DSL. Pushing job creation into a script allows you to automate and standardize your Jenkins installation, unlike anything possible before.'

Using the Job DSL plugin for Jenkins.

- a pipeline script allows you to Programmatically control what a job does,
- A job dsl script allows you to Programmatically create jobs themselves.

@jeffreymckenzie



This is not built into Jenkins,
But you can install it through plugin Manager.

@jeffreymckenzie

New Item [Jenkins] x +

localhost:8080/view/all/newJob

Jenkins Jeff McKenzie | log out

Jenkins > All >

Enter an item name

reindeer-dsl

» 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.

OK

Let's see how that works --
You create a DSL job as a freestyle project

@jeffreymckenzie

The screenshot shows the Jenkins web interface for configuring a job named 'reindeer-dsl'. The browser address bar indicates the URL is 'localhost:8080/view/Jenkins/job/reindeer-dsl/configure'. The Jenkins logo and user 'Jeff McKenzie' are visible in the top navigation bar. The configuration page has several tabs: 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. The 'General' tab is active, showing a 'Description' field with the text 'Creates all reindeer jobs.' and a '[Plain text] Preview' link below it. A list of checkboxes is present, including 'Discard old builds', 'GitHub project', 'This project is parameterized', 'Throttle builds', 'Use another job as a template', and 'Allow this job to be used as a template'. At the bottom, there are 'Save' and 'Apply' buttons, and a 'Disable this project' link.

reindeer-dsl Config [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-dsl/configure

Jenkins

search Jeff McKenzie log out

Jenkins > Jenkins > reindeer-dsl >

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

Description Creates all reindeer jobs.

[Plain text] [Preview](#)

☐ Discard old builds

☐ GitHub project

☐ This project is parameterized

☐ Throttle builds

☐ Use another job as a template

☐ Allow this job to be used as a template

[Disable this project](#)

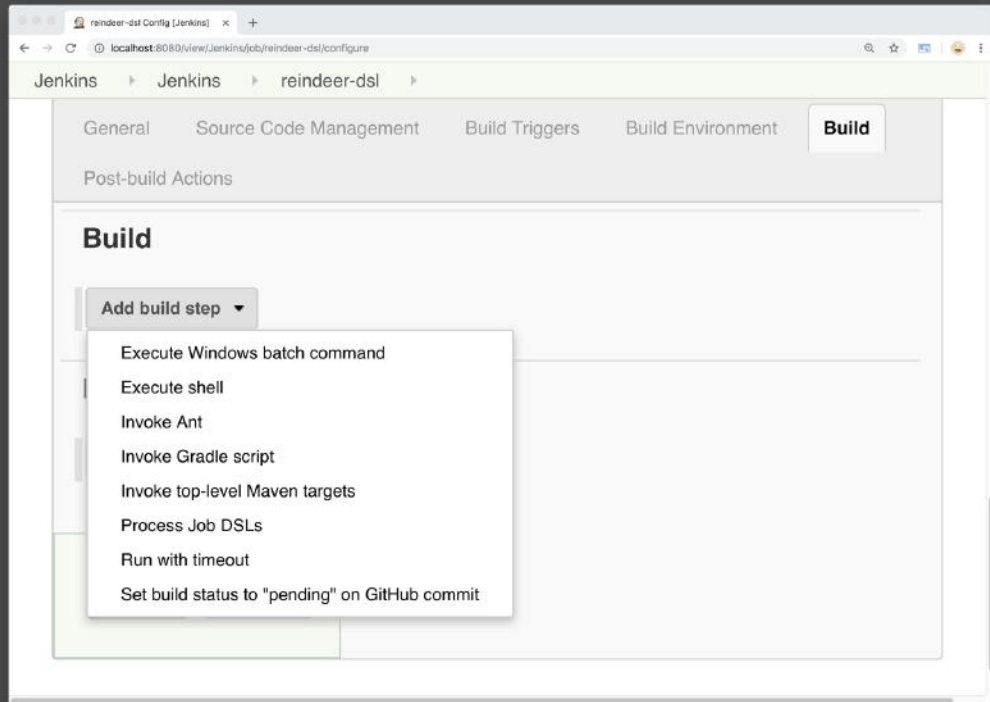
Save Apply

is necessary

Give description –

Creates all reindeer jobs

@jeffreymckenzie



Then we go down to build and select the step
Process job DSLs –

Again, this will only show up
if you have the plugin installed

@jeffreymckenzie

reindeer-dsl Config [Jenkins] x +

localhost:8080/view/Jenkins/job/reindeer-dsl/configure

Jenkins » Jenkins » reindeer-dsl »

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

Build

Process Job DSLs [X]

See [Job DSL API](#) for syntax reference.

☐ Use the provided DSL script

☒ Look on Filesystem

DSL Scripts [v] [?]

Ignore missing files: ☐ [?]

Use Groovy Sandbox: ☐ [?]

Action for existing jobs and views: ☐ Ignore changes [?]

Action for removed jobs: [?]

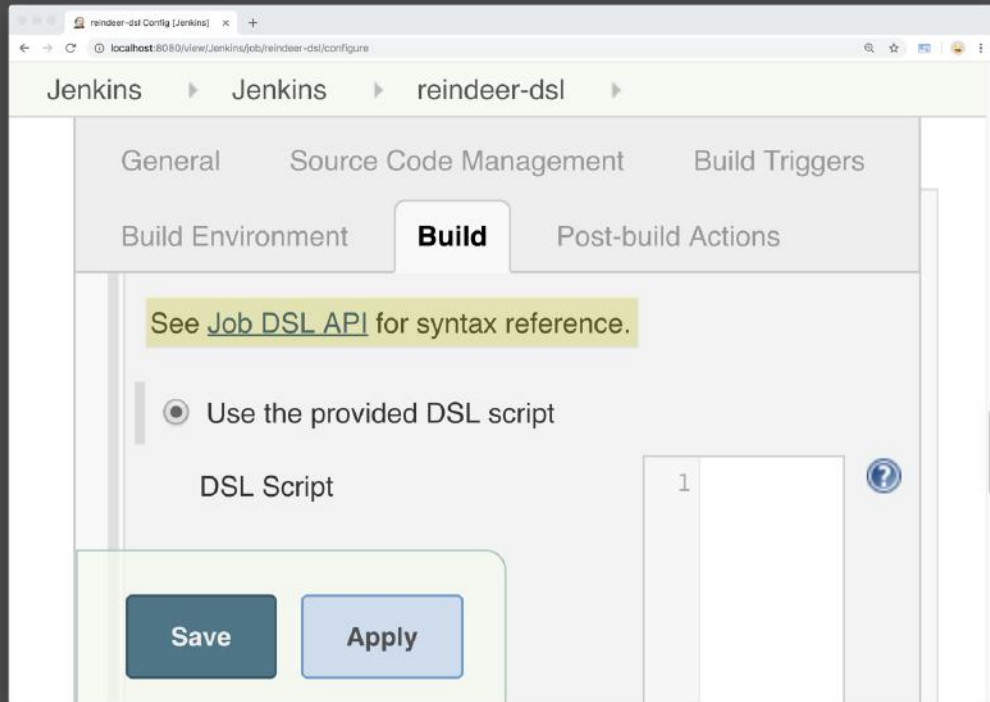
[?]

Have a couple options –

We can pull from filesystem,
Which means we can version control this –

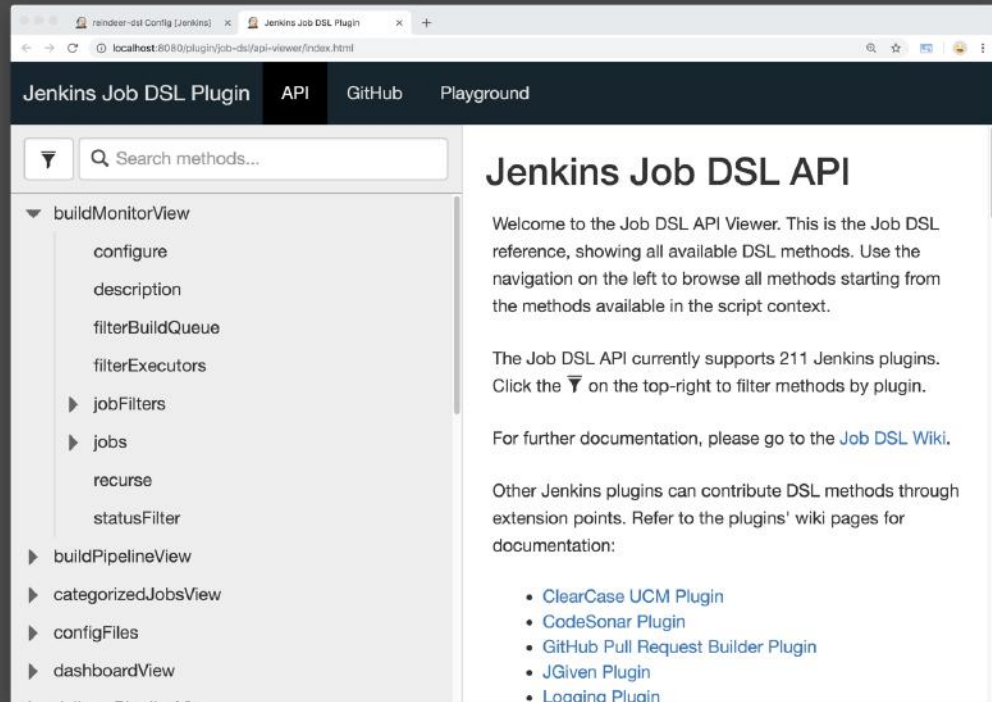
Or we can enter a script right here,
Which we will do for demo purposes.

@jeffreymckenzie



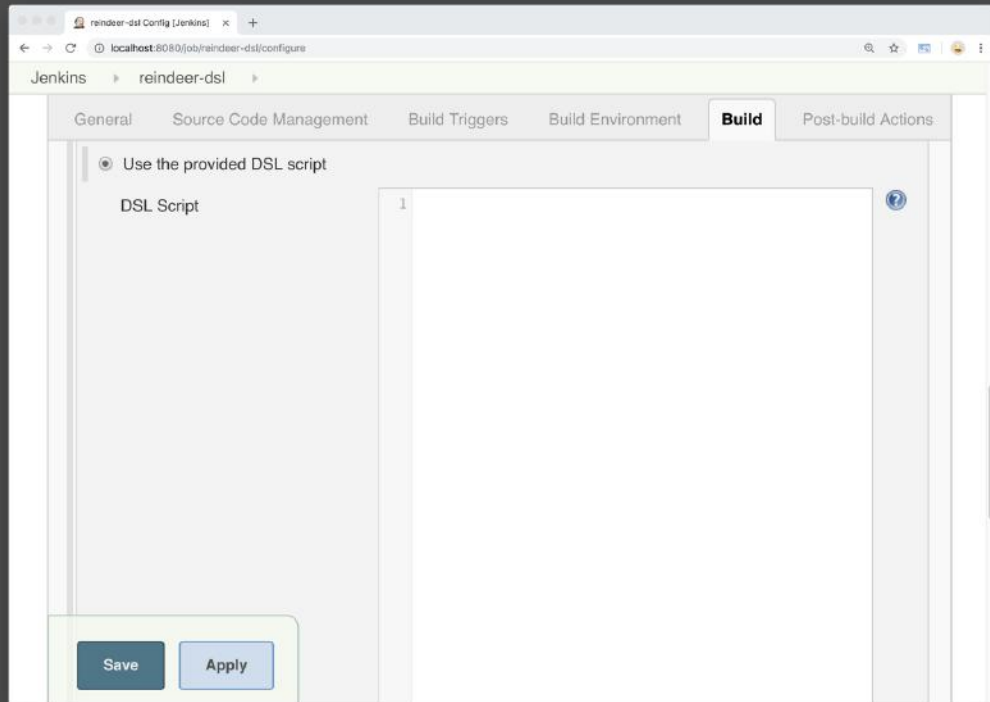
And on that page there's a link to the
Job DSL API that you can visit
To learn more about the scripting syntax

@jeffreymckenzie



It has a really nice viewer
That you can search
To find out more about how
To create Jenkins items
Through the DSL

@jeffreymckenzie



So here's the space to put our script
Before we do this,
We need to make one adjustment to our pipeline script....

```
#!/groovy
@Library('jenkins-lib') _

options = [
    greeting: 'Hello',
    reindeer: 'Rudolph'
]
reindeer(options)
```

Going to do 2 things –

1. Put this in source control under the name “Jenkinsfile”
2. Switch out the hard-coded values to use parameters

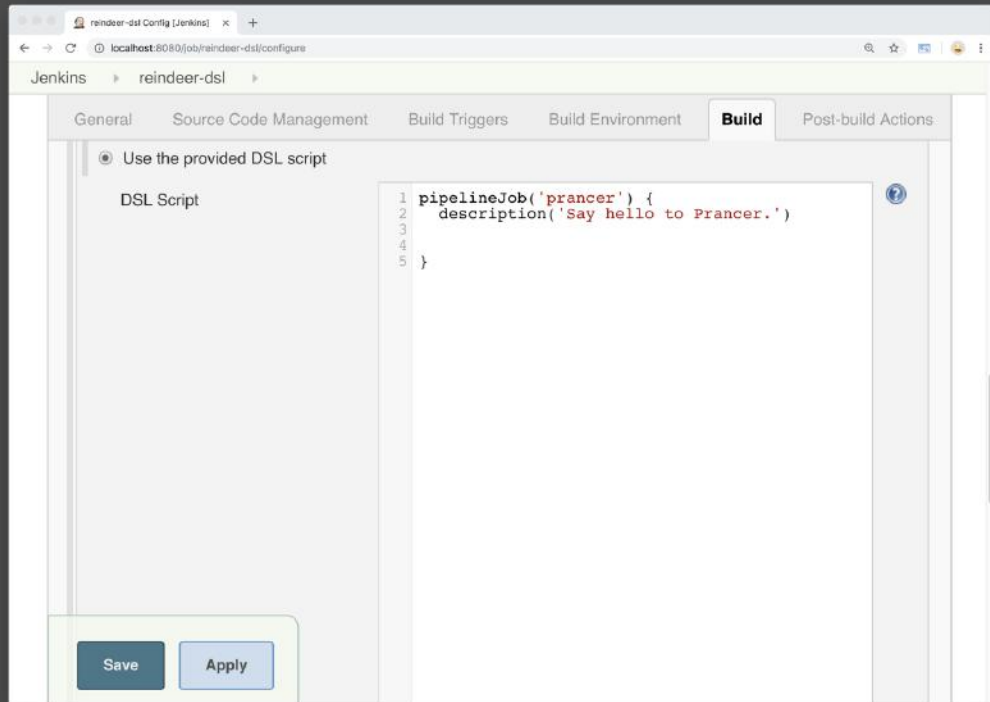

```
- Jenkinsfile
#!groovy
@Library('jenkins-lib') _

options = [
    greeting: GREETING,
    reindeer: REINDEER
]
reindeer(options)
```

Otherwise, it's the same.

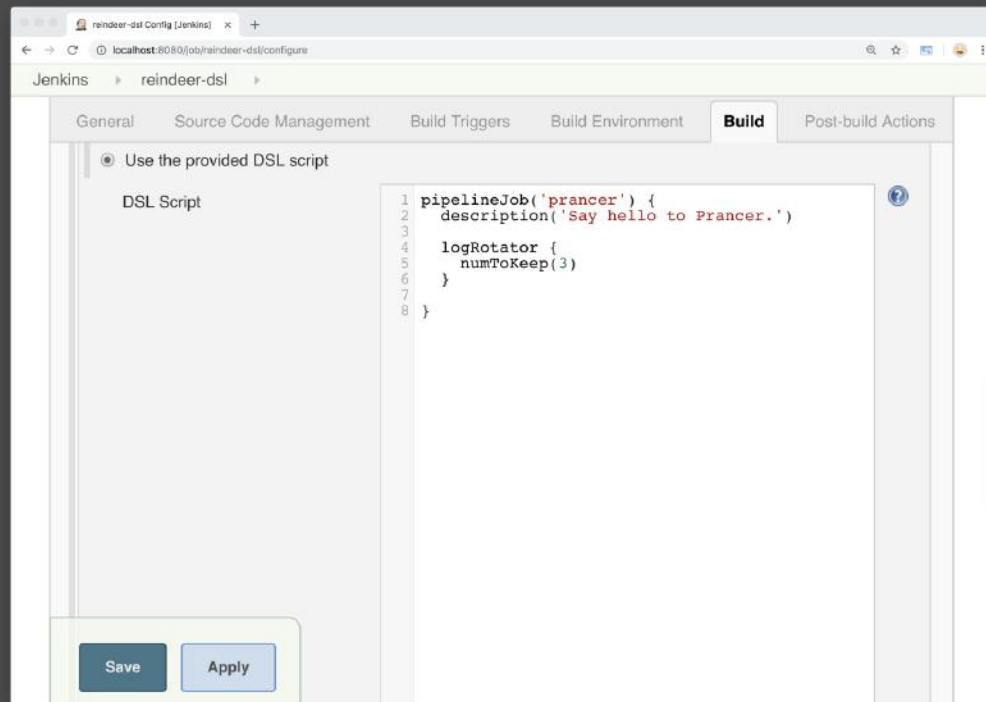
Lets go back to our dsl project and start to build it

@jeffreymckenzie



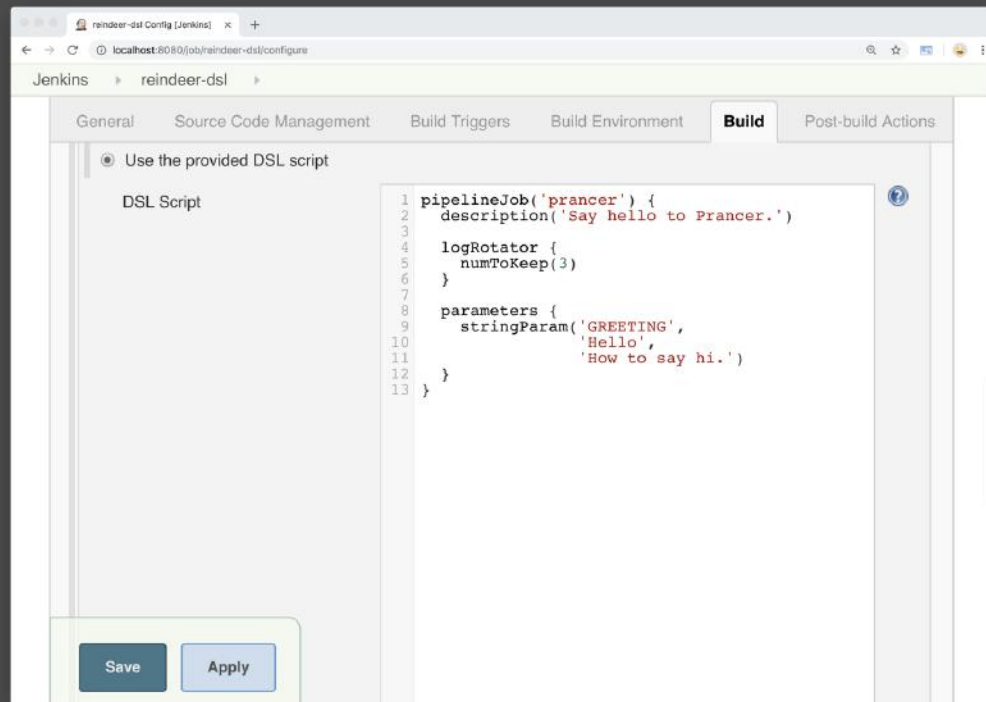
To create a pipeline job or project,
We simply use the pipelineJob command,
With the name of the project –
We'll call this one Prancer.
Then we give it a description.

@jeffreymckenzie



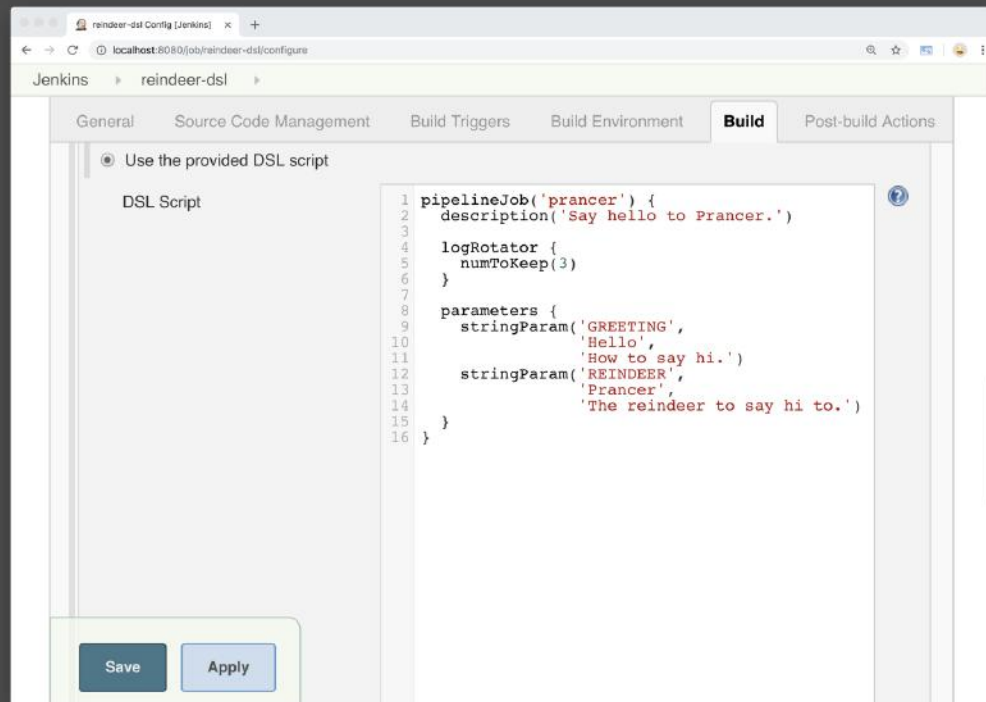
We add our build/log history –
We want to keep the last 3 build logs.

@jeffreymckenzie



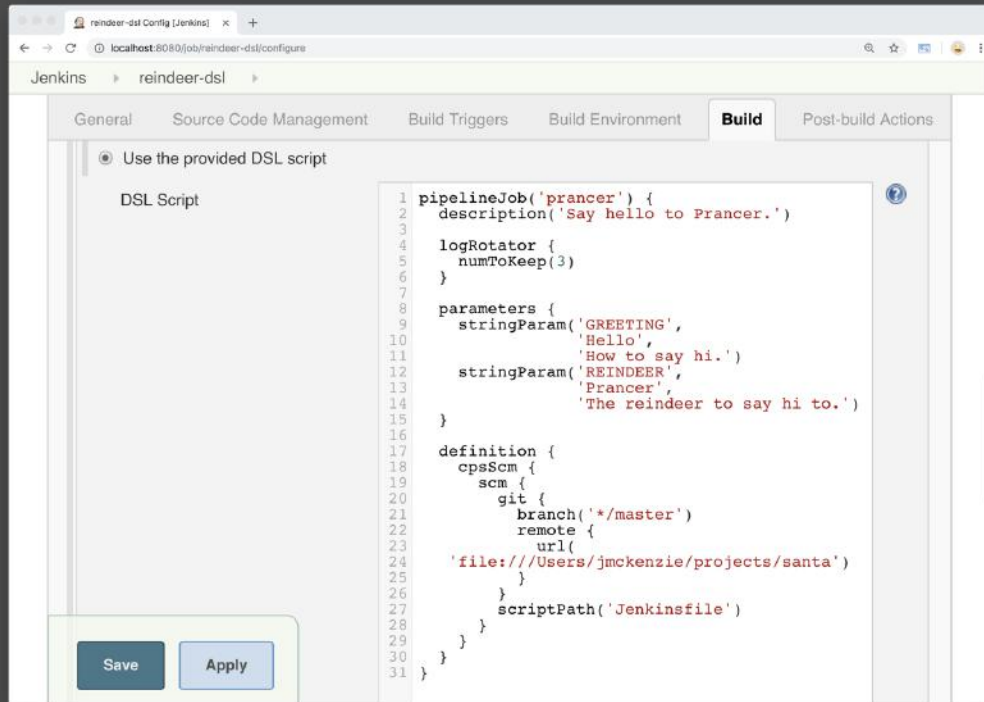
We add our GREETING parameter

@jeffreymckenzie



Then our reindeer parameter

@jeffreymckenzie

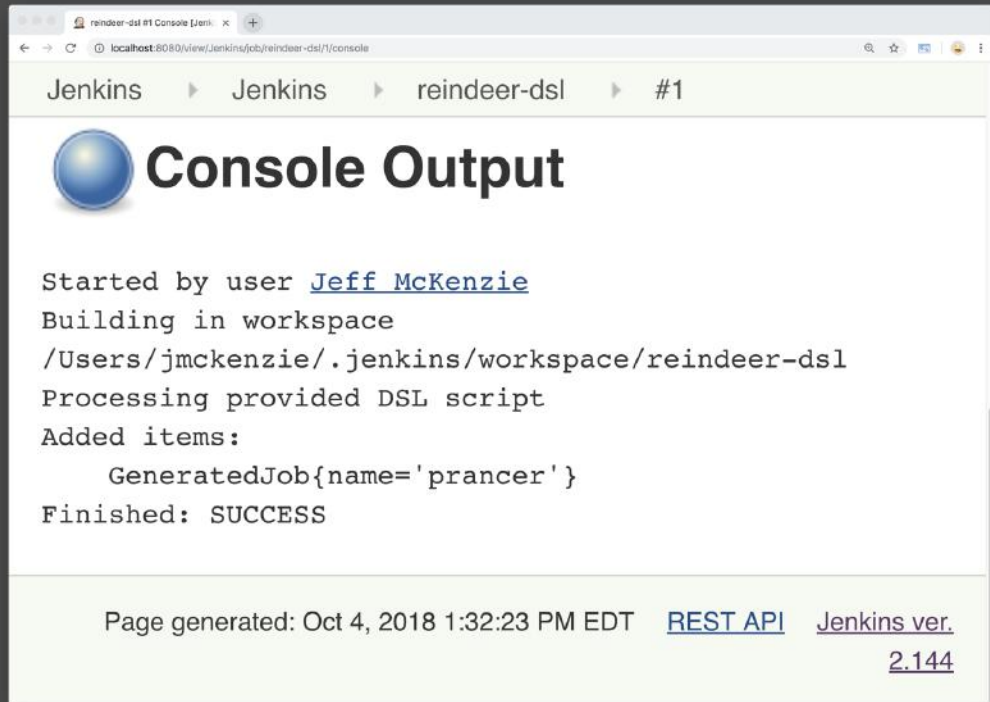


```
1 pipelineJob('prancer') {
2   description('Say hello to Prancer.')
3
4   logRotator {
5     numToKeep(3)
6   }
7
8   parameters {
9     stringParam('GREETING',
10      'Hello',
11      'How to say hi.')
12     stringParam('REINDEER',
13      'Prancer',
14      'The reindeer to say hi to.')
15   }
16
17   definition {
18     cpsScm {
19       scm {
20         git {
21           branch('*/master')
22           remote {
23             url(
24               'file:///Users/jmckenzie/projects/santa')
25           }
26           scriptPath('Jenkinsfile')
27         }
28       }
29     }
30   }
31 }
```

And then finally the build definition itself –
This just tells Jenkins what pipeline script to pull
And which git repo to get it from.

So let's run this and see what it does...

@jeffreymckenzie



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/view/Jenkins/job/reindeer-dsl/1/console'. The breadcrumb navigation shows 'Jenkins > Jenkins > reindeer-dsl > #1'. The main heading is 'Console Output' with a blue sphere icon. The console text reads: 'Started by user [Jeff McKenzie](#)', 'Building in workspace /Users/jmckenzie/.jenkins/workspace/reindeer-dsl', 'Processing provided DSL script', 'Added items:', 'GeneratedJob{name='prancer'}', and 'Finished: SUCCESS'. The footer indicates 'Page generated: Oct 4, 2018 1:32:23 PM EDT' and provides links for 'REST API' and 'Jenkins ver. 2.144'.

reindeer-dsl #1 Console [Jenkins]

localhost:8080/view/Jenkins/job/reindeer-dsl/1/console

Jenkins > Jenkins > reindeer-dsl > #1

Console Output

Started by user [Jeff McKenzie](#)
Building in workspace
/Users/jmckenzie/.jenkins/workspace/reindeer-dsl
Processing provided DSL script
Added items:
GeneratedJob{name='prancer'}
Finished: SUCCESS

Page generated: Oct 4, 2018 1:32:23 PM EDT [REST API](#) [Jenkins ver. 2.144](#)

So it tells us we've created
the prancer job successfully..
Let's go take a look.

@jeffreymckenzie

The screenshot shows the Jenkins web interface for the 'Reindeer DSL' job. The page includes a sidebar with navigation links such as 'New Item', 'People', 'Build History', 'Edit View', 'Delete View', 'Manage Jenkins', 'My Views', 'Credentials', and 'New View'. The main content area displays a table of jobs with the following columns: 'S' (Status), 'W' (Web icon), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. The 'prancer' job is highlighted, showing a last success of 'N/A' and a last failure of 'N/A'. Below the table, there is a 'Build Queue' section showing 'No builds in the queue' and a 'Build Executor Status' section showing '1 Idle' and '2 Idle'.

S	W	Name	Last Success	Last Failure	Last Duration
		prancer	N/A	N/A	N/A
		reindeer-dsl	2 min 42 sec - #1	N/A	0.19 sec

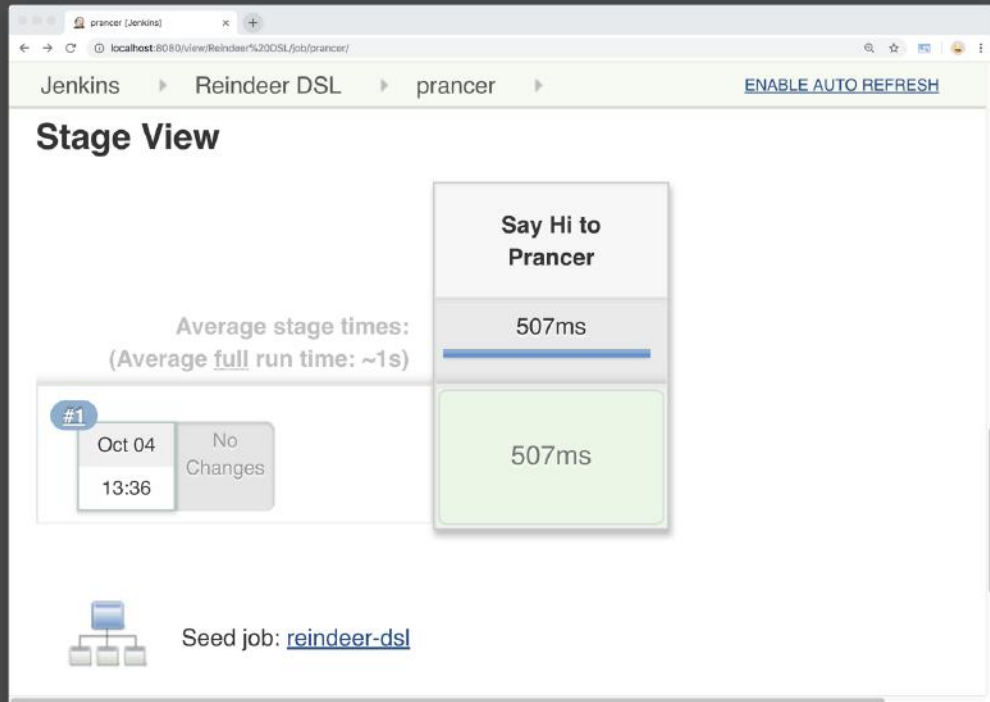
Icon: [S](#) [M](#) [L](#)

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

Page generated: Oct 4, 2018 1:34:54 PM EDT [REST API](#) Jenkins ver. 2.144

So if we go back to the main list,
You can see the prancer job,
Which was created programmatically
Through the DSL Script.
Let's run it.

@jeffreymckenzie

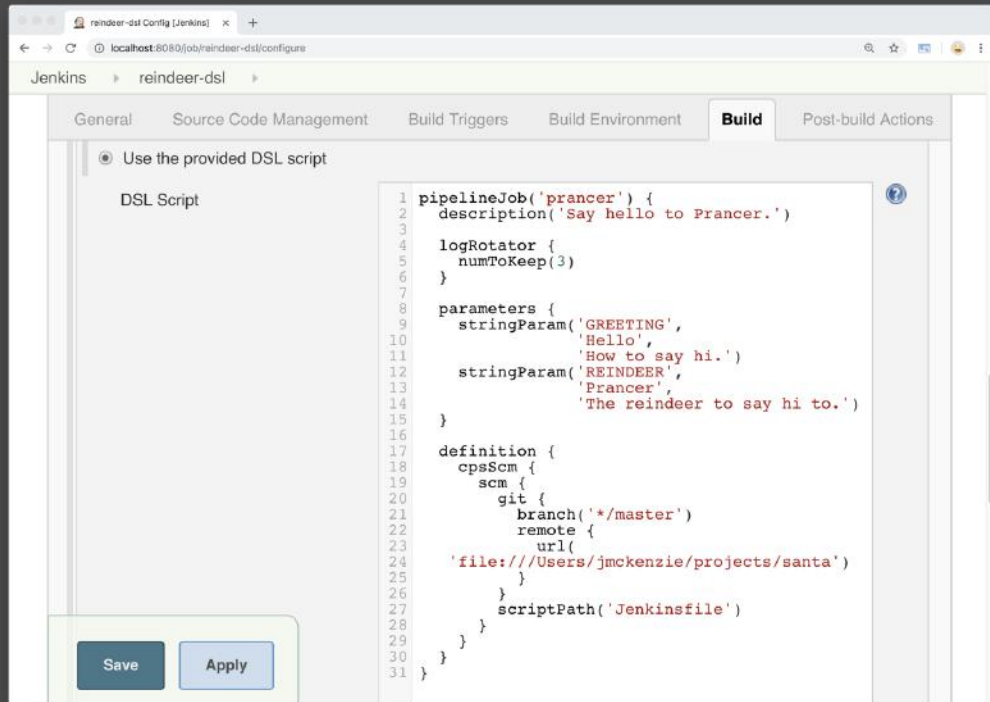


If we look at the stage view,
we can see that it runs successfully,
But this time saying hi to prance

So this is all good, but we've only created one job.

The power of Job DSL is that it allows you
To add scripting to declaration

@jeffreymckenzie

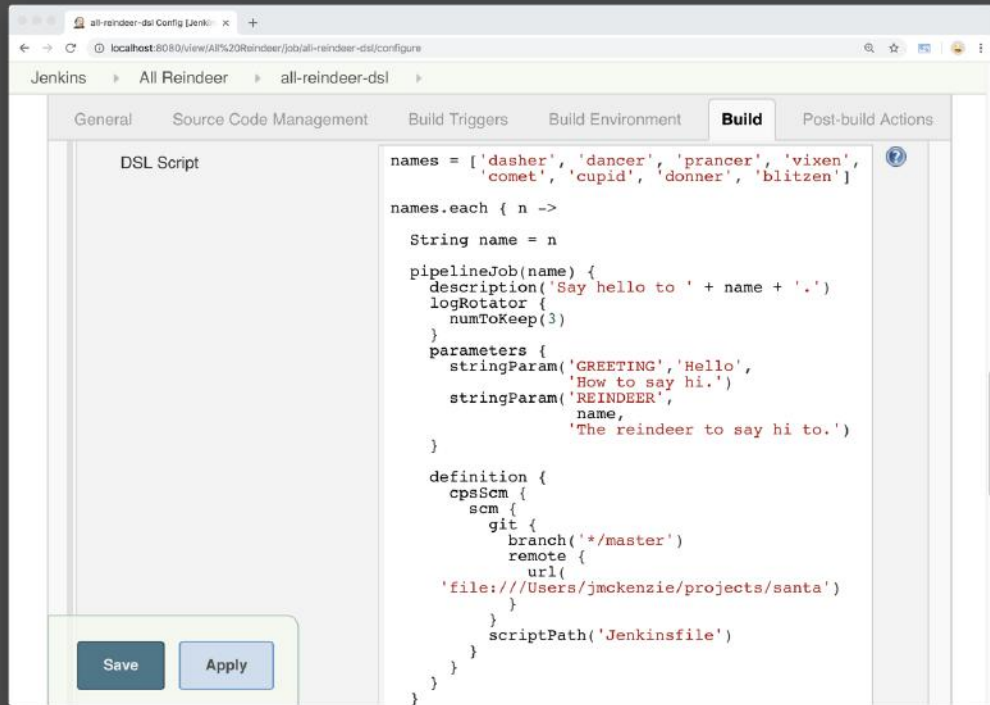


```
1 pipelineJob('prancer') {
2   description('Say hello to Prancer.')
3
4   logRotator {
5     numToKeep(3)
6   }
7
8   parameters {
9     stringParam('GREETING',
10      'Hello',
11      'How to say hi.')
12     stringParam('REINDEER',
13      'Prancer',
14      'The reindeer to say hi to.')
15   }
16
17   definition {
18     cpsScm {
19       scm {
20         git {
21           branch('*/master')
22           remote {
23             url(
24               'file:///Users/jmckenzie/projects/santa')
25           }
26         }
27       }
28     }
29   }
30 }
31 }
```

So here's our original DSL script –
Creates the prancer job.

Let's modify that slightly.

@jeffreymckenzie



```
names = ['dasher', 'dancer', 'prancer', 'vixen',
        'comet', 'cupid', 'donner', 'blitzen']

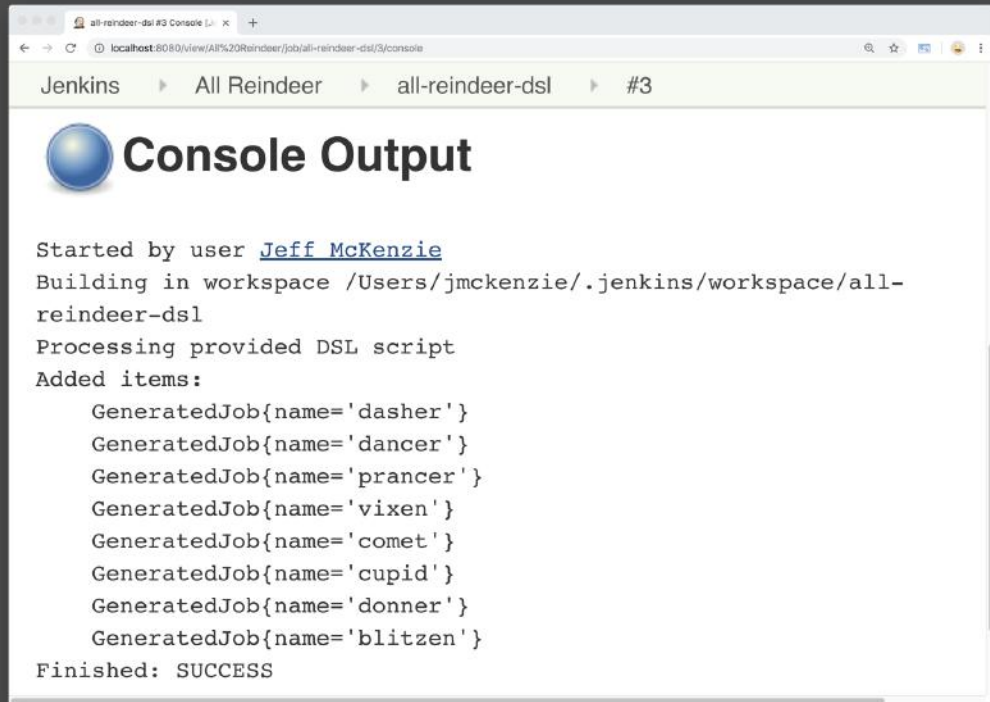
names.each { n ->
    String name = n

    pipelineJob(name) {
        description('Say hello to ' + name + '.')
        logRotator {
            numToKeep(3)
        }
        parameters {
            stringParam('GREETING', 'Hello',
                'How to say hi.')
            stringParam('REINDEER',
                name,
                'The reindeer to say hi to.')
        }
    }

    definition {
        cpsScm {
            scm {
                git {
                    branch('*/master')
                    remote {
                        url(
                            'file:///Users/jmckenzie/projects/santa')
                    }
                }
                scriptPath('Jenkinsfile')
            }
        }
    }
}
```

At the top we are going to add an array
Of reindeer names,
Loop through each of those names,
And create a job for each name,
Inserting the name where we need it.
Let's run this.

@jeffreymckenzie

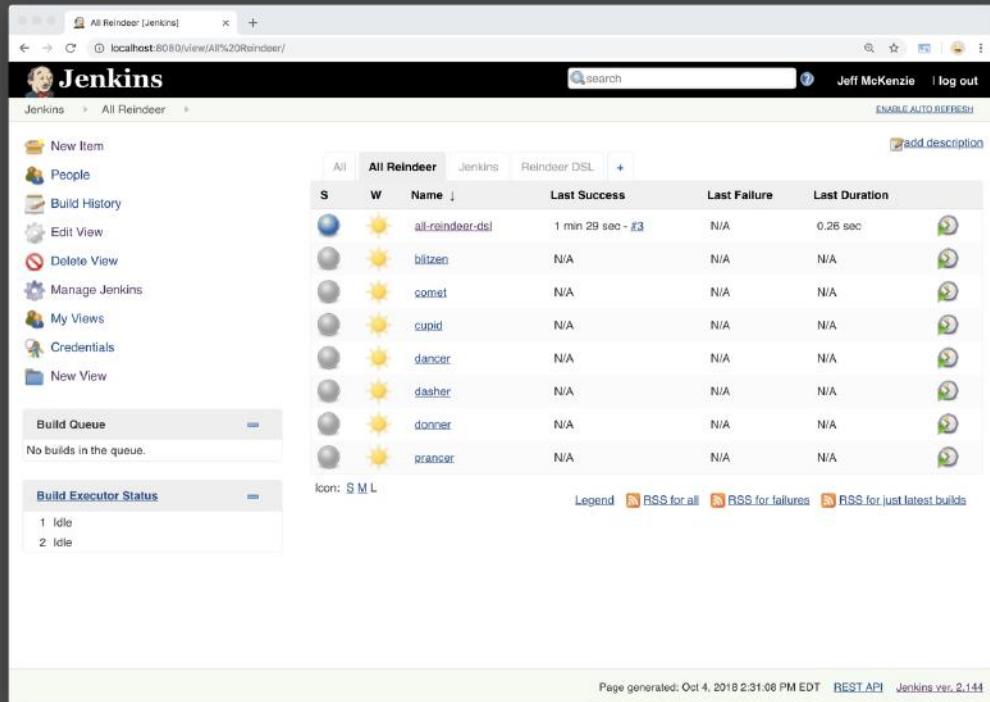


The screenshot shows a web browser window displaying the Jenkins console output for a job named 'all-reindeer-dsl'. The breadcrumb navigation at the top reads 'Jenkins > All Reindeer > all-reindeer-dsl > #3'. The main heading is 'Console Output' with a blue sphere icon. The output text is as follows:

```
Started by user Jeff McKenzie  
Building in workspace /Users/jmckenzie/.jenkins/workspace/all-reindeer-dsl  
Processing provided DSL script  
Added items:  
    GeneratedJob{name='dasher'}  
    GeneratedJob{name='dancer'}  
    GeneratedJob{name='prancer'}  
    GeneratedJob{name='vixen'}  
    GeneratedJob{name='comet'}  
    GeneratedJob{name='cupid'}  
    GeneratedJob{name='donner'}  
    GeneratedJob{name='blitzen'}  
Finished: SUCCESS
```

So this created all the jobs –
Let's look at the main page.

@jeffreymckenzie



The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and the user name 'Jeff McKenzie' with a 'log out' link. The main content area displays a list of jobs under the 'All Reindeer' view. The jobs are listed in a table with columns for status, name, last success, last failure, and last duration. The jobs are: all-reindeer-dsl, blitzen, comet, cupid, dancer, dasher, donner, and prancer. The 'all-reindeer-dsl' job has a last success of '1 min 29 sec - #3' and a last duration of '0.26 sec'. The other jobs have a last success of 'N/A' and a last duration of 'N/A'. The left sidebar contains links for 'New Item', 'People', 'Build History', 'Edit View', 'Delete View', 'Manage Jenkins', 'My Views', 'Credentials', and 'New View'. The bottom of the page shows the page generation date and time: 'Page generated: Oct 4, 2018 2:31:08 PM EDT'.

S	W	Name	Last Success	Last Failure	Last Duration
		all-reindeer-dsl	1 min 29 sec - #3	N/A	0.26 sec
		blitzen	N/A	N/A	N/A
		comet	N/A	N/A	N/A
		cupid	N/A	N/A	N/A
		dancer	N/A	N/A	N/A
		dasher	N/A	N/A	N/A
		donner	N/A	N/A	N/A
		prancer	N/A	N/A	N/A

There they are.

Now we can drill down into any of these

And run them, and they should

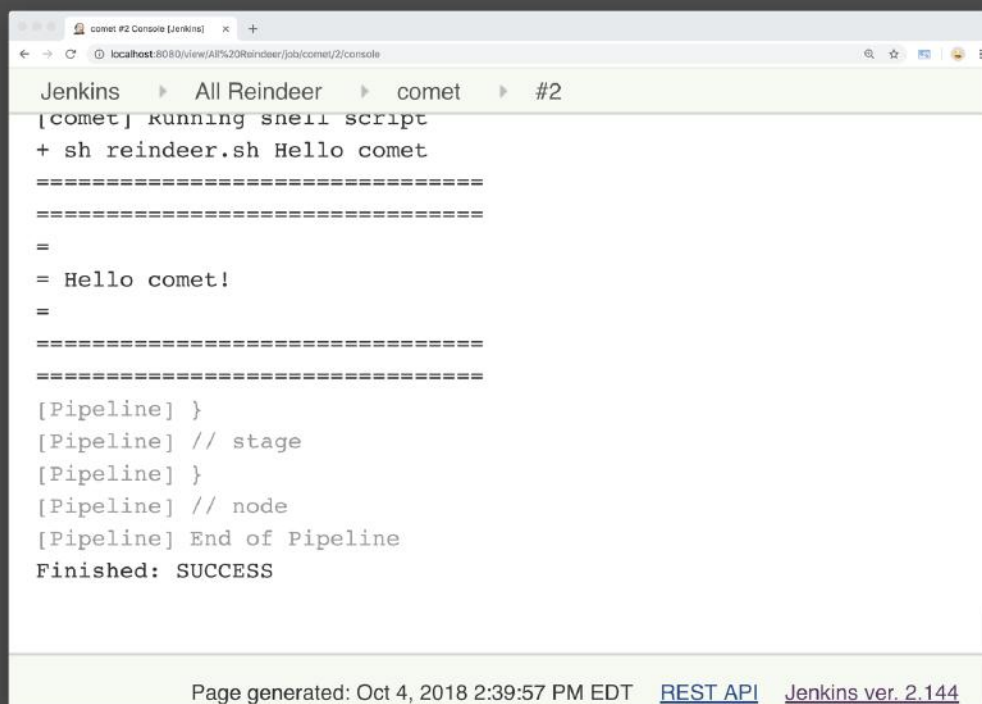
Function just as if we created them manually...

@jeffreymckenzie

The screenshot shows the Jenkins web interface for a pipeline named 'comet'. The breadcrumb navigation at the top reads 'Jenkins > All Reindeer > comet'. A link 'ENABLE AUTO REFRESH' is in the top right. The main heading is 'Pipeline comet', followed by the text 'Say hello to comet.'. On the right, there is a link 'edit description' with a pencil icon and a blue button labeled 'Disable Project'. Below this, on the left, is a 'Recent Changes' link with a notepad icon. The 'Stage View' section contains a light blue box with the message: 'No data available. This Pipeline has not yet run.'

So let's look at comet –
Hasn't run yet...

@jeffreymckenzie



The screenshot shows a web browser window displaying the Jenkins console for a job named 'comet'. The breadcrumb navigation at the top reads 'Jenkins > All Reindeer > comet > #2'. The console output shows the pipeline running a shell script 'reindeer.sh' with the argument 'Hello comet'. The output is formatted with a box around the message 'Hello comet!'. Below the output, the pipeline structure is shown with stages and nodes, ending with 'Finished: SUCCESS'. At the bottom of the console, it says 'Page generated: Oct 4, 2018 2:39:57 PM EDT' followed by links for 'REST API' and 'Jenkins ver. 2.144'.

```
Jenkins > All Reindeer > comet > #2
[comet] Running shell script
+ sh reindeer.sh Hello comet
=====
=====
=
= Hello comet!
=
=====
=====

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

Page generated: Oct 4, 2018 2:39:57 PM EDT REST API Jenkins ver. 2.144
```

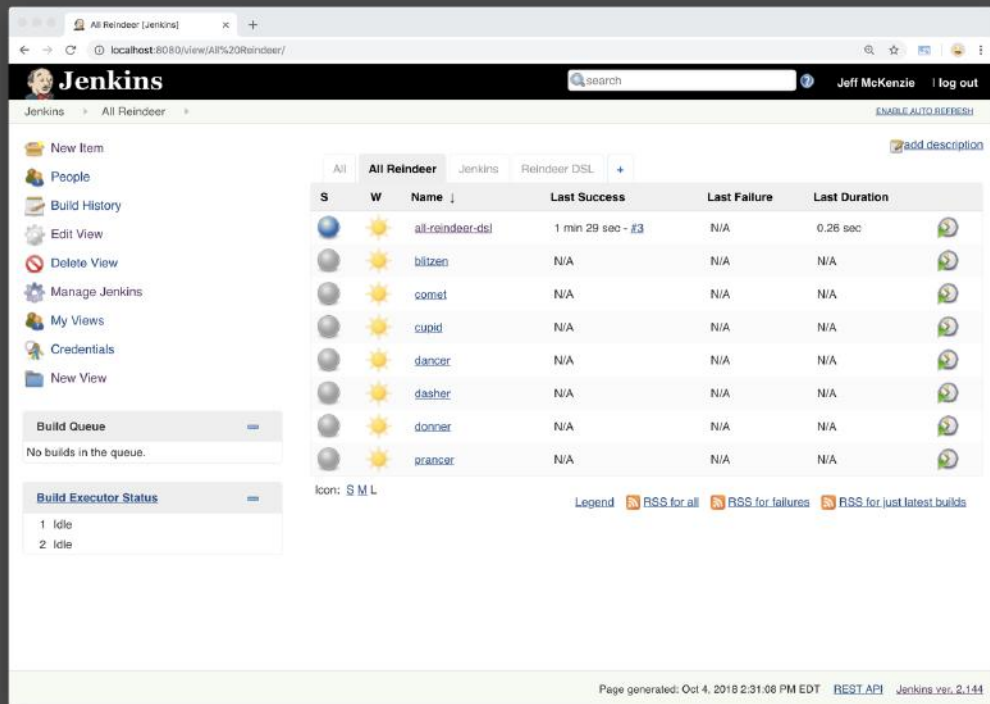
we run it, and we get the expected result.

@jeffreymckenzie



So a quick recap of really how little effort it takes
[what santa has been able to do]
To get to this end state...

@jeffreymckenzie



The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and the user name 'Jeff McKenzie' with a 'log out' link. The left sidebar contains various navigation options: 'New Item', 'People', 'Build History', 'Edit View', 'Delete View', 'Manage Jenkins', 'My Views', 'Credentials', and 'New View'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '1 Idle' and '2 Idle'). The main content area displays a table of jobs under the 'All Reindeer' view. The table has columns for 'S' (status), 'W' (weather icon), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. The jobs listed are 'all-reindeer-dsl', 'blitzen', 'comet', 'cupid', 'dancer', 'dasher', 'donner', and 'prancer'. The 'all-reindeer-dsl' job shows a last success of '1 min 29 sec - #3' and a last duration of '0.26 sec'. Below the table, there is a legend for RSS feeds: 'RSS for all', 'RSS for failures', and 'RSS for just latest builds'. The footer indicates the page was generated on 'Oct 4, 2018 2:31:08 PM EDT' and provides links for 'REST API' and 'Jenkins ver. 2.144'.

S	W	Name	Last Success	Last Failure	Last Duration
		all-reindeer-dsl	1 min 29 sec - #3	N/A	0.26 sec
		blitzen	N/A	N/A	N/A
		comet	N/A	N/A	N/A
		cupid	N/A	N/A	N/A
		dancer	N/A	N/A	N/A
		dasher	N/A	N/A	N/A
		donner	N/A	N/A	N/A
		prancer	N/A	N/A	N/A

...of having all these jobs generated.

```
- reindeer.groovy
#!/usr/bin/env groovy
def call(Map<String, Object> options) {
    try {
        node {
            String greeting = options."greeting".toString()
            String reindeer = options."reindeer".toString()
            stage("Say Hi to ${reindeer}") {
                checkout scm: [
                    $class: 'GitSCM',
                    branches: [[name: '*/master']],
                    userRemoteConfigs: [[url:
                        'file:///Users/jmckenzie/projects/santa']]
                sh "sh reindeer.sh ${greeting} ${reindeer}"
            }
        }
    }
    catch (Throwable err) {
        throw err
    }
}
```

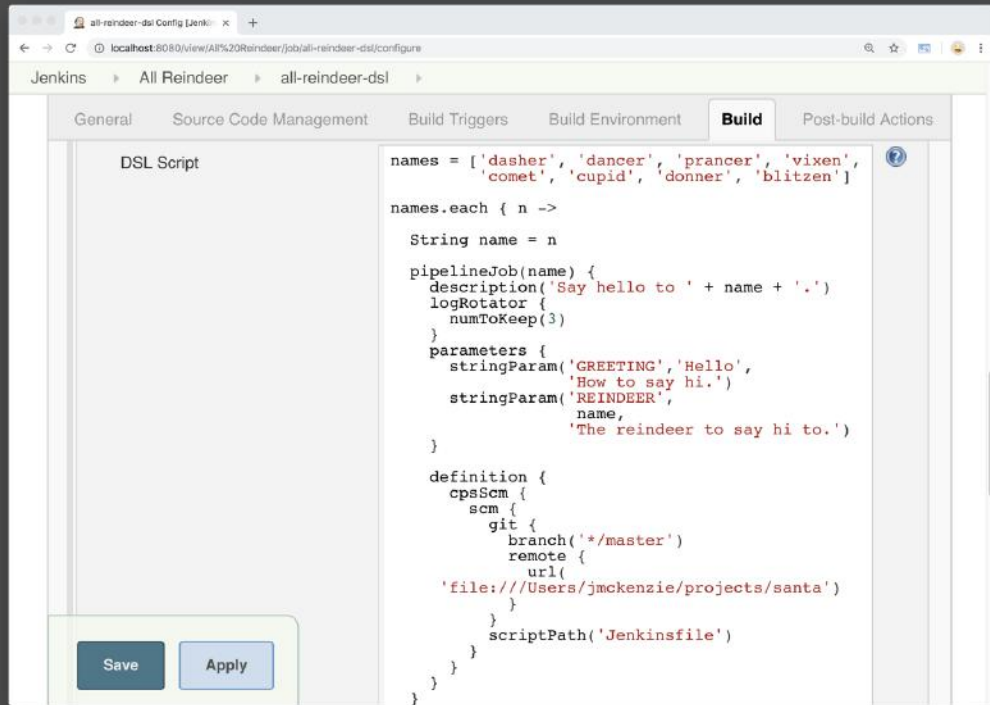
We have our pipeline code –
One instance –
Shared across all of Jenkins,
Versioned, In source control

```
- Jenkinsfile
#!groovy
@Library('jenkins-lib') _

options = [
    greeting: GREETING,
    reindeer: REINDEER
]
reindeer(options)
```

We have our Jenkinsfile,
Again, once instance of it,
Versioned, and in source control

@jeffreymckenzie




```
names = ['dasher', 'dancer', 'prancer', 'vixen',
        'comet', 'cupid', 'donner', 'blitzen']

names.each { n ->
    String name = n
    pipelineJob(name) {
        description('Say hello to ' + name + '.')
        logRotator {
            numToKeep(3)
        }
        parameters {
            stringParam('GREETING', 'Hello',
                'How to say hi.')
            stringParam('REINDEER',
                name,
                'The reindeer to say hi to.')
        }
    }
    definition {
        cpsScm {
            scm {
                git {
                    branch('*/master')
                    remote {
                        url(
                            'file:///Users/jmckenzie/projects/santa')
                    }
                }
                scriptPath('Jenkinsfile')
            }
        }
    }
}
```

And finally, our one DSL job and script
So to scale this out further
You only really need to add an entry in that array.

So all of that in about the same amount
Of code that you probably throw away each day.



Automating Your Automation

The Care and Feeding of Jenkins

Jeff McKenzie • jeff.mckenzie@insight.com • [@jeffreymckenzie](https://twitter.com/jeffreymckenzie)

??????