Cookbooks

Organizing Recipes

CHEF

Objectives

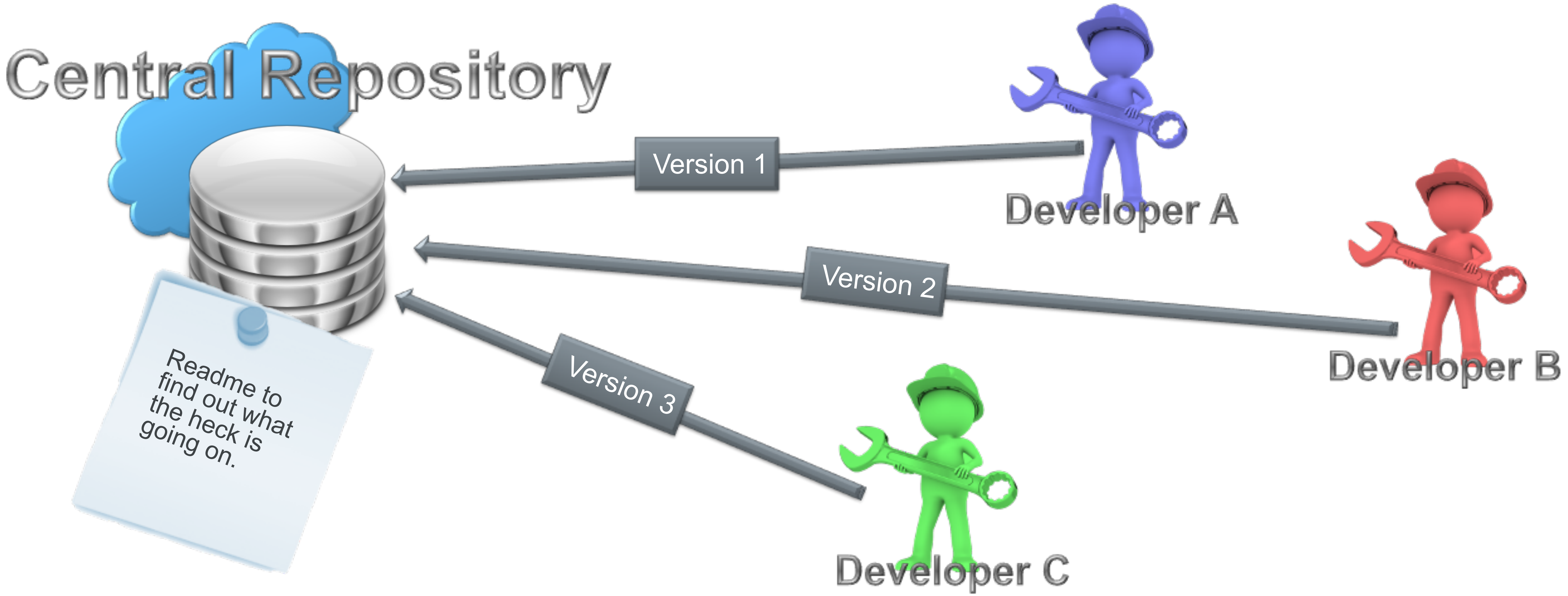
After completing this module, you should be able to:

- ➢ Modify a recipe
- ➢ Use version control
- ➢ Generate a Chef cookbook
- ➢ Define a Chef recipe that sets up a web server

Questions You May Have

1. Thinking about the workstation recipe, could we do something like that for a web server?
2. Is there a way to package up recipes you create with a version number (and maybe a README)?
3. I think `chef` is able to generate something called a cookbook. Shouldn't we start thinking about some version control so we don't lose all our hard work?

Collaboration and Version Control



Git Version Control

git is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows.

We will be using **git** throughout the rest of this course.

Test Git Account

Email address: chefaspe@gmail.com

Password: chefaspe1

Git username: chefaspe

Password: chefaspe1

```
git config --global user.email "chefaspe@gmail.com"
git config --global user.name "Chef Aspe Shekhar"
```

GL: Create a Cookbook

How are we going to manage this file? Does it need a README?

Objective:

- ❑ Use chef to generate a cookbook
- ❑ Move the setup recipe into the new cookbook
- ❑ Add the new cookbook to version control

Cookbooks

A Chef cookbook is the fundamental unit of configuration and policy distribution.

Each cookbook defines a scenario, such as everything needed to install and configure MySQL, and then it contains all of the components that are required to support that scenario.

Read the first three paragraphs here: http://docs.chef.io/cookbooks.html

Working within Home Directory

```
$ cd ~
```

GL: Create a Cookbooks Directory

```
$ mkdir cookbooks
```

What can 'chef' do?

```
$ chef --help
```

```
UsaGL:
    chef -h/--help
    chef -v/--version
    chef command [arguments...] [options...]

Available Commands:
    exec          Runs the command in context of the embedded ruby
    gem           Runs the `gem` command in context of the embedded ruby
    generate      Generate a new app, cookbook, or component
    shell-init    Initialize your shell to use ChefDK as your primary ruby
    install       Install cookbooks from a Policyfile and generate a locked cookboo...
    update        Updates a Policyfile.lock.json with latest run_list and cookbooks
```

What Can 'chef generate' Do?

```
$ chef generate --help
```

```
UsaGL: chef generate GENERATOR [options]


Available generators:
  app         Generate an application repo
  cookbook    Generate a single cookbook
  recipe      Generate a new recipe
  attribute   Generate an attributes file
  template    Generate a file template
  file        Generate a cookbook file
  lwrp        Generate a lightweight resource/provider
  repo        Generate a Chef policy repository
  policyfile  Generate a Policyfile for use with the install/push commands
  generator   Copy ChefDK's generator cookbook so you can customize it
```

GL: Let's Create a Cookbook

```
$ chef generate cookbook cookbooks/workstation
```

```
Generating cookbook workstation
- Ensuring correct cookbook file content
- Committing cookbook files to git
- Ensuring delivery configuration
- Ensuring correct delivery build cookbook content
- Adding delivery configuration to feature branch
- Adding build cookbook to feature branch
- Merging delivery content feature branch to master

Your cookbook is ready. Type `cd cookbooks/workstation` to enter it.

There are several commands you can run to get started locally developing and
testing your cookbook.
Type `delivery local --help` to see a full list.
```

GL: The Cookbook Has a README

```
$ tree cookbooks/workstation
```

```
workstation
├── Berksfile       Changed to Policyfile.rb in Chef 15
├── chefignore
├── metadata.rb
├── README.md
├── recipes
│   └── default.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
| ...
6 directories, 8 files
```

CONCEPT

README.md

The description of the cookbook's features written in Markdown.

http://daringfireball.net/projects/markdown/syntax

GL: The Cookbook Has Some Metadata

```
$ tree cookbooks/workstation
```

```
workstation
├── Berksfile
├── chefignore
├── metadata.rb
├── README.md
├── recipes
│   └── default.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
| ...
6 directories, 8 files
```

Changed to Policyfile.rb in Chef 15

DOCS

metadata.rb

Every cookbook requires a small amount of metadata. Metadata is stored in a file called metadata.rb that lives at the top of each cookbook's directory.

http://docs.chef.io/config_rb_metadata.html

CHEF

GL: Let's Take a Look at the Metadata

```
$ cat cookbooks/workstation/metadata.rb
```

```
name             'workstation'
maintainer       'The Authors'
maintainer_email 'you@example.com'
license          'all_rights'
description      'Installs/Configures workstation'
long_description 'Installs/Configures workstation'
version          '0.1.0'

# If you upload to Supermarket you should set this so your cookbook
# gets a `View Issues` link
# issues_url 'https://github.com/<insert_org_here>/workstation/issues' if
respond_to?(:issues_url)
```

CHEF

GL: The Cookbook Has a Folder for Recipes

```
$ tree cookbooks/workstation
```

```
workstation
├── Berksfile
├── chefignore
├── metadata.rb
├── README.md
├── recipes
│   └── default.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
| ...
6 directories, 8 files
```

GL: The Cookbook Has a Default Recipe

```
$ cat cookbooks/workstation/recipes/default.rb
```

```
# Cookbook Name:: workstation
# Recipe:: default
#
# Copyright (c) 2016 The Authors, All Rights Reserved.
```

GL: Create a Cookbook

How are we going to manage this file? Does it need a README?

Objective:

- ✓ Use chef to generate a cookbook
- ❑ Move the setup recipe into the new cookbook
- ❑ Add the new cookbook to version control

GL: Copy the Recipe into the Cookbook

```
$ mv setup.rb cookbooks/workstation/recipes
```

GL: Verify the Cookbook has the Recipe

```
$ tree cookbooks/workstation
```

```
workstation
├── Berksfile
├── chefignore
├── metadata.rb
├── README.md
├── recipes
|    └── default.rb
|    └── setup.rb
├── spec
|    ├── spec_helper.rb
|    └── unit
| ...
6 directories, 9 files
```

Lab: Apply the Server Recipe

```
$ sudo chef-client -z
cookbooks/workstation/recipes/server.rb
```

```
Converging 2 resources

…………..
```

Group Exercise: Version Control

This is a probably a good point to capture the initial state of our cookbook.

Objective:

- ✓ Use chef to generate a cookbook
- ✓ Move the setup recipe into the new cookbook
- ❑ Add the new cookbook to version control

GL: Move into the Cookbook Directory

```
$ cd cookbooks/workstation
```

REMOTE

GL: Initialize the Directory as a git Repository

```
$ git init
```

```
Reinitialized existing Git repository in /home/chef/cookbooks/workstation/.git/
```

GL: Use 'git add' to Stage Files to be Committed

```
$ git add .
```

REMOTE

CONCEPT

Staging Area

The staging area has a file, generally contained in your Git directory, that stores information about what will go into your next commit.

It’s sometimes referred to as the “index”, but it’s also common to refer to it as the staging area.

http://git-scm.com/book/en/v2/Getting-Started-Git-Basics

GL: Use 'git status' to View the Staged Files

```
$ git status
```

```
On branch master


Initial commit


Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   .gitignore
        new file:   .kitchen.yml
        new file:   Berksfile
        new file:   README.md
        new file:   chefignore
        new file:   metadata.rb
```

GL: Use 'git commit' to Save the Staged Changes

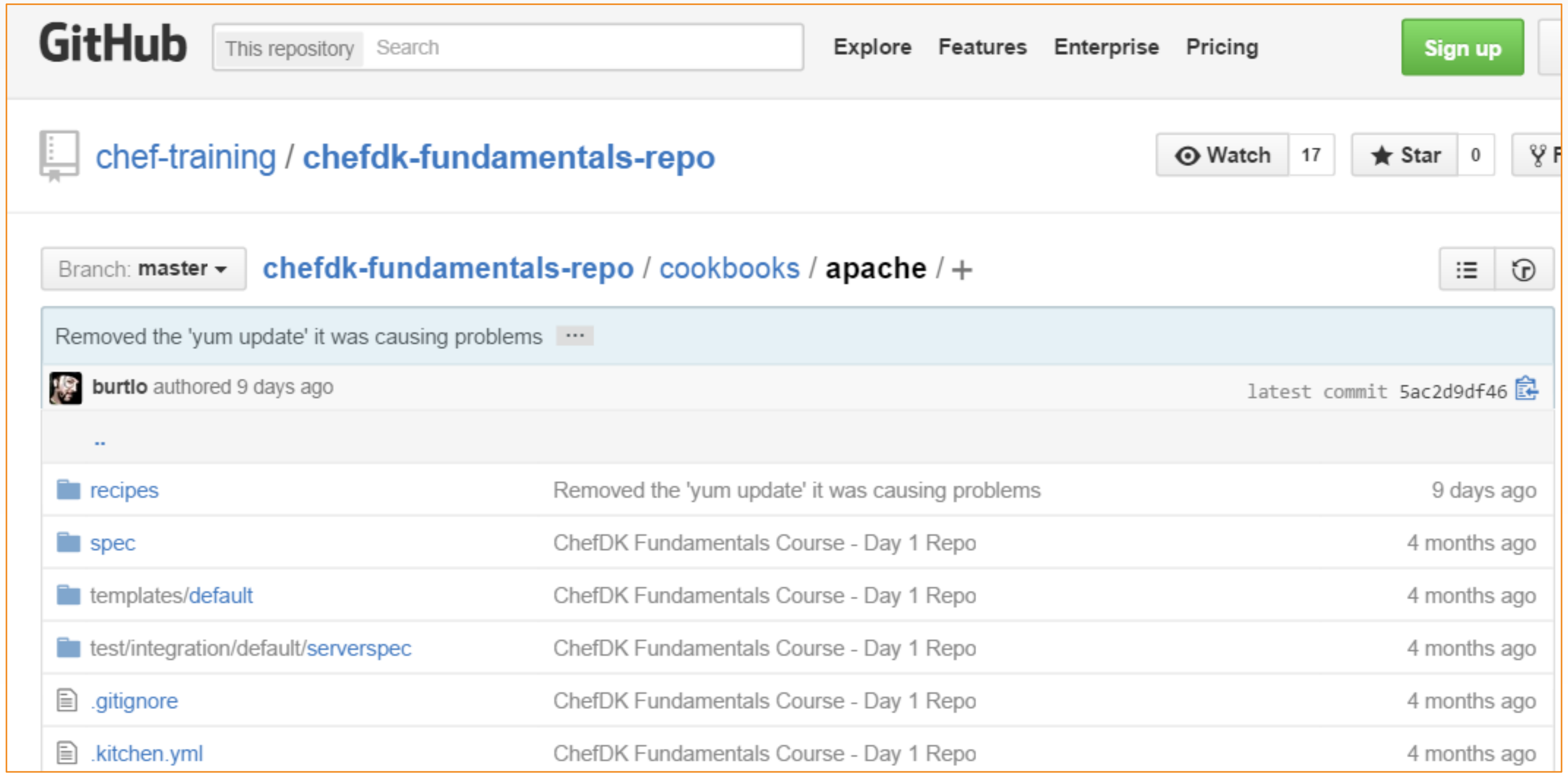
```
$ git commit -m "Initial commit"
```

```
[master (root-commit) 73b39cb] Initial commit
 Committer: ChefDK User <chef@ip-172-31-14-46.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author  ...
```

Git Version Control

If you use git versioning you should ultimately push the local git repository to a shared remote git repository.

In this way others could collaborate with you from a centralized location.



chef-training / chefdk-fundamentals-repo

Branch: master ▾ chefdk-fundamentals-repo / cookbooks / apache / +

Removed the 'yum update' it was causing problems ...

burtlo authored 9 days ago latest commit 5ac2d9df46

..

recipes	Removed the 'yum update' it was causing problems	9 days ago
spec	ChefDK Fundamentals Course - Day 1 Repo	4 months ago
templates/default	ChefDK Fundamentals Course - Day 1 Repo	4 months ago
test/integration/default/serverspec	ChefDK Fundamentals Course - Day 1 Repo	4 months ago
.gitignore	ChefDK Fundamentals Course - Day 1 Repo	4 months ago
.kitchen.yml	ChefDK Fundamentals Course - Day 1 Repo	4 months ago

GL: Return to the Home Directory

```
$ cd ~
```

Lab: Setting up a Web Server

- ❑ Use `chef generate` to create a cookbook named "apache".
- ❑ Write and apply a recipe named **`"server.rb"`** with the policy:

  The package named 'httpd' is installed.

  The file named '/var/www/html/index.html' is created with the content '<h1>Hello, world!</h1>'

  The service named 'httpd' is started and enabled.
- ❑ Apply the recipe with chef-client
- ❑ Verify the site is available by running **`curl localhost`**

Lab: Create a Cookbook

```
$ chef generate cookbook cookbooks/apache
```

```
Generating cookbook apache
- Ensuring correct cookbook file content
- Committing cookbook files to git
- Ensuring delivery configuration
- Ensuring correct delivery build cookbook content
- Adding delivery configuration to feature branch
- Adding build cookbook to feature branch
- Merging delivery content feature branch to master

Your cookbook is ready. Type `cd cookbooks/apache` to enter it.

There are several commands you can run to get started locally developing and
testing your cookbook.
Type `delivery local --help` to see a full list.
```

Lab: Create a Cookbook

```
$ chef generate recipe cookbooks/apache server
```

```
Compiling Cookbooks...
Recipe: code_generator::recipe
  * directory[cookbooks/apache/spec/unit/recipes] action create (up to date)
  * cookbook_file[cookbooks/apache/spec/spec_helper.rb] action create_if_missing
 (up to date)
  * template[cookbooks/apache/spec/unit/recipes/server_spec.rb] action
create_if_missing
    - create new file cookbooks/apache/spec/unit/recipes/server_spec.rb
    - update content in file cookbooks/apache/spec/unit/recipes/server_spec.rb
from none to a43970
    (diff output suppressed by config)
  * template[cookbooks/apache/recipes/server.rb] action create
    - create new file cookbooks/apache/recipes/server.rb
    - update content in file cookbooks/apache/recipes/server.rb from none to
3d6b92
```

Lab: Create the Server Recipe

```
~/cookbooks/apache/recipes/server.rb
```

```
package 'httpd'

file '/var/www/html/index.html' do
  content '<h1>Hello, world!</h1>'
end

service 'httpd' do
  action [:enable, :start]
end
```

Lab: Apply the Server Recipe

```
$ sudo chef-client -z cookbooks/apache/recipes/server.rb
```

```
Converging 3 resources
Recipe: @recipe_files::/home/chef/cookbooks/apache/recipes/server.rb
  * yum_package[httpd] action install
    - install version 2.2.15-47.el6.centos.3 of package httpd
  * file[/var/www/html/index.html] action create
    - create new file /var/www/html/index.html
    - update content in file /var/www/html/index.html from none to 17d291
    --- /var/www/html/index.html    2016-02-24 21:41:45.494844958 +0000
    +++ /var/www/html/.index.html20160224-10036-6y8on7    2016-02-24
21:41:45.493844958 +0000
    @@ -1 +1,2 @@
    +<h1>Hello, world!</h1>
  * service[httpd] action enable
    - enable service service[httpd]
```

Lab: Verify That the Website is Available

```
$ curl localhost
```

```
<h1>Hello, world!</h1>
```

CONCEPT

Linting with cookstyle and foodcritic

`cookstyle` and `foodcritic` are two easy-to-use linting tools that are included with Chef Workstation.

CHEF™

CONCEPT

Linting with cookstyle

You can simply run `cookstyle` from the directory where your recipe resides and it will indicate any syntax offenses.

https://docs.chef.io/cookstyle.html

CHEF™

Example: Running cookstyle in workstation Cookbook

```
$ cookstyle
```

```
Inspecting 9 files
..C......
Offenses:

recipes/server.rb:15:1: C: 1 trailing blank lines detected.
recipes/server.rb:15:1: C: Trailing whitespace detected.

9 files inspected, 2 offenses detected
```

In this example, we ran `cookstyle` from within the recipes directory where we created the server.rb recipe. `cookstyle` has detected a trailing blank line, although it's not a critical error.

Example: Running cookstyle in a Cookbook

```
$ cookstyle
```

```
Inspecting 6 files
..C...


Offenses:


recipes/default.rb:13:27: C: Trailing whitespace detected.
  action [:start, :enable]
                          ^
```

In this example from a different recipe, `cookstyle' even shows via the caret where the trailing whitespace exists.

Example: `cookstyle –a`

```
$ cookstyle -a
```

```
Inspecting 2 files
CC
Offenses:
default.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
disable-uac.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
...
2 files inspected, 10 offenses detected, 8 offenses corrected
```

`cookstyle –a` will detect and auto-correct syntax errors within the directory you are located. You may want to save a copy of the original file (or utilize git) before running `cookstyle –a` just in case you don't like the correction(s) made.

GL: Run cookstyle at the Cookbook Level

```
> cd ~\cookbooks\workstation
> cookstyle
```

```
Inspecting 7 files
CCCC.CC
Offenses:
metadata.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
name 'workstation' ...
^^^^^^^^^^^^^^^^
Policyfile.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
# Policyfile.rb - Describe how you want Chef Infra Client to build your system. ...
....
7 files inspected, 14 offenses detected recipes/default.rb:1:1: C: Layout/EndOfLine: Carriage
return character detected.
```

GL: Run cookstyle at the Recipes Level

```
> cd ~\cookbooks\workstation\recipes
> cookstyle
```

```
disable-uac.rb:13:5: C: Style/HashSyntax: Use the new Ruby 1.9 hash syntax.
    :name => 'ConsentPromptBehaviorAdmin',
    ^^^^^^^^
...
disable-uac.rb:15:5: C: Style/TrailingCommaInHashLiteral: Put a comma after the last item of
a multiline hash.
    :data => 0
    ^^^^^^^^^^


2 files inspected, 10 offenses detected
```

GL: Run cookstyle on an Individual File

```
> cookstyle default.rb
```

```
Inspecting 1 file

C
Offenses:

default.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
# ...
^
1 file inspected, 1 offense detected
```

CONCEPT

Foodcritic

Foodcritic is a static linting tool that analyzes all of the Ruby code that is authored in a cookbook against a number of rules, and then returns a list of violations.

Because Foodcritic is a static linting tool, using it is fast. The code in a cookbook is read, broken down, and then compared to Foodcritic rules.

Foodcritic

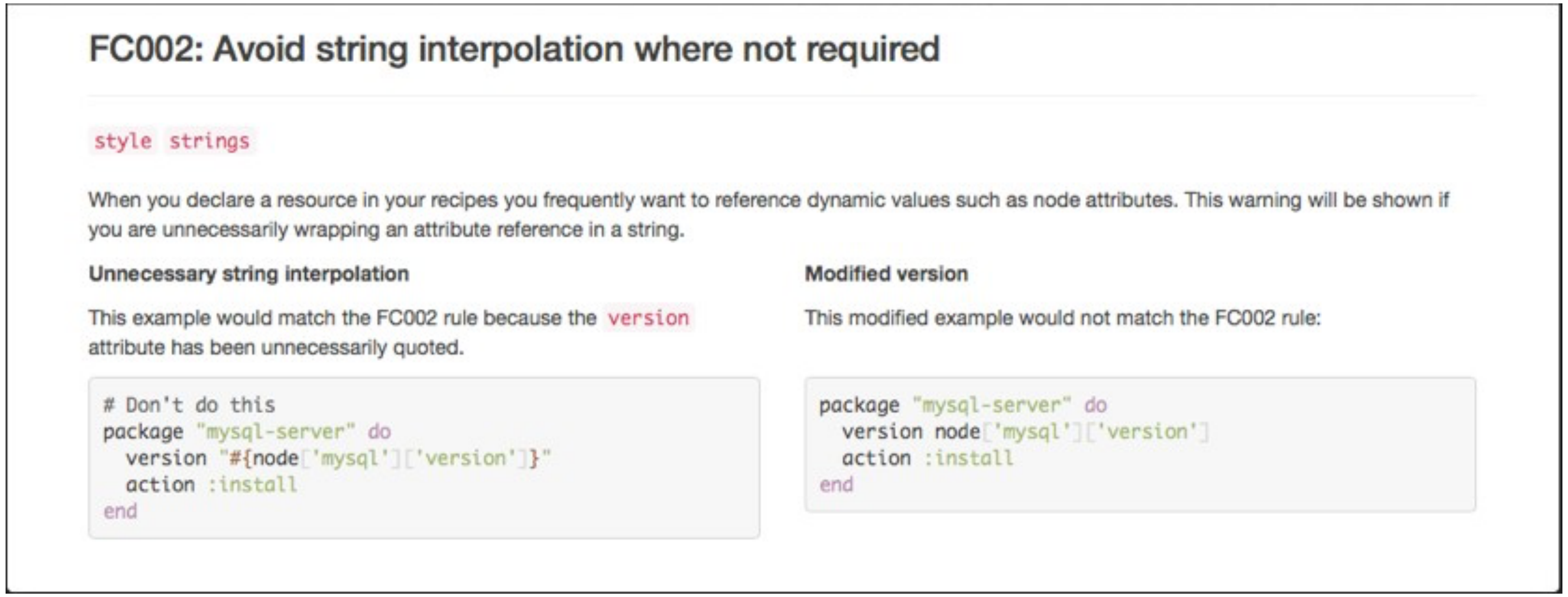
Use foodcritic to check cookbooks for common problems:

- Style
- Correctness
- Syntax
- Best practices
- Common mistakes
- Deprecations

https://docs.chef.io/foodcritic.html
http://www.foodcritic.io/

Rules and Tags

- All rules are numbered, e.g.
  - *FC002 - "Avoid string interpolation where not required"*
  - *FC034 - "Unused template variables"*
- Rules can be categorized and tagged, e.g.
  - 'style'
  - 'portability'
  - ...



FC002: Avoid string interpolation where not required

`style` `strings`

When you declare a resource in your recipes you frequently want to reference dynamic values such as node attributes. This warning will be shown if you are unnecessarily wrapping an attribute reference in a string.

Unnecessary string interpolation

This example would match the FC002 rule because the `version` attribute has been unnecessarily quoted.

```
# Don't do this
package "mysql-server" do
  version "#{node['mysql']['version']}"
  action :install
end
```

Modified version

This modified example would not match the FC002 rule:

```
package "mysql-server" do
  version node['mysql']['version']
  action :install
end
```

Foodcritic

- There are 50+ rules in Foodcritic; your cookbooks should pass them (or have a good reason for not)
- You can write your own rules
- Extra community-contributed rules: http://www.foodcritic.io/#extra-rules

GL: Run `foodcritic` on the workstation Cookbook

```
> cd ~\cookbooks\workstation
> foodcritic .
```

```
Checking 3 files
x..
FC008: Generated cookbook metadata needs updating: ./metadata.rb:2
FC008: Generated cookbook metadata needs updating: ./metadata.rb:3
FC064: Ensure issues_url is set in metadata: ./metadata.rb:1
FC065: Ensure source_url is set in metadata: ./metadata.rb:1
FC067: Ensure at least one platform supported in metadata: ./metadata.rb:1
FC078: Ensure cookbook shared under an OSI-approved open source license:
./metadata.rb:1
FC093: Generated README text needs updating: ./README.md:1
```

GL: Run `foodcritic` on the myiis Cookbook

```
> cd ~\cookbooks\myiis
> foodcritic .
```

```
Checking 4 files
x..x
FC008: Generated cookbook metadata needs updating: ./metadata.rb:2
FC008: Generated cookbook metadata needs updating: ./metadata.rb:3
FC064: Ensure issues_url is set in metadata: ./metadata.rb:1
FC065: Ensure source_url is set in metadata: ./metadata.rb:1
FC067: Ensure at least one platform supported in metadata: ./metadata.rb:1
FC078: Ensure cookbook shared under an OSI-approved open source license:
./metadata.rb:1
FC093: Generated README text needs updating: ./README.md:1
```

CHEF™

GL: Run `foodcritic` on a Single Error Code

```
> foodcritic . -t FC008
```

```
Checking 4 files


x...


FC008: Generated cookbook metadata needs updating: ./metadata.rb:2
FC008: Generated cookbook metadata needs updating: ./metadata.rb:3
```

Best Practice: Run Foodcritic Before Each Commit

Always check in correct code!

Make Foodcritic a part of your build pipeline with commit hooks or other methods.

COMMIT

GL: Commit Your Work

```
$ cd cookbooks/apache
$ git add .
$ git commit -m "Initial commit"
```

Lab: 60 minutes

https://github.com/shekhar2010us/chef-essentials-repo-15/blob/master/labs/chapter%203.md

CHEF

Discussion

What file would you read first when examining a cookbook?

What other recipes might you include in the apache or workstation cookbook?

Can resources accept multiple actions?

How often would you commit changes with version control?

Q&A

What questions can we answer for you?

- Cookbooks
- Versions
- Version control

CHEF™