Creating a Web Server Cookbook

Chef DK, Test Kitchen, Resources, and more

Objectives

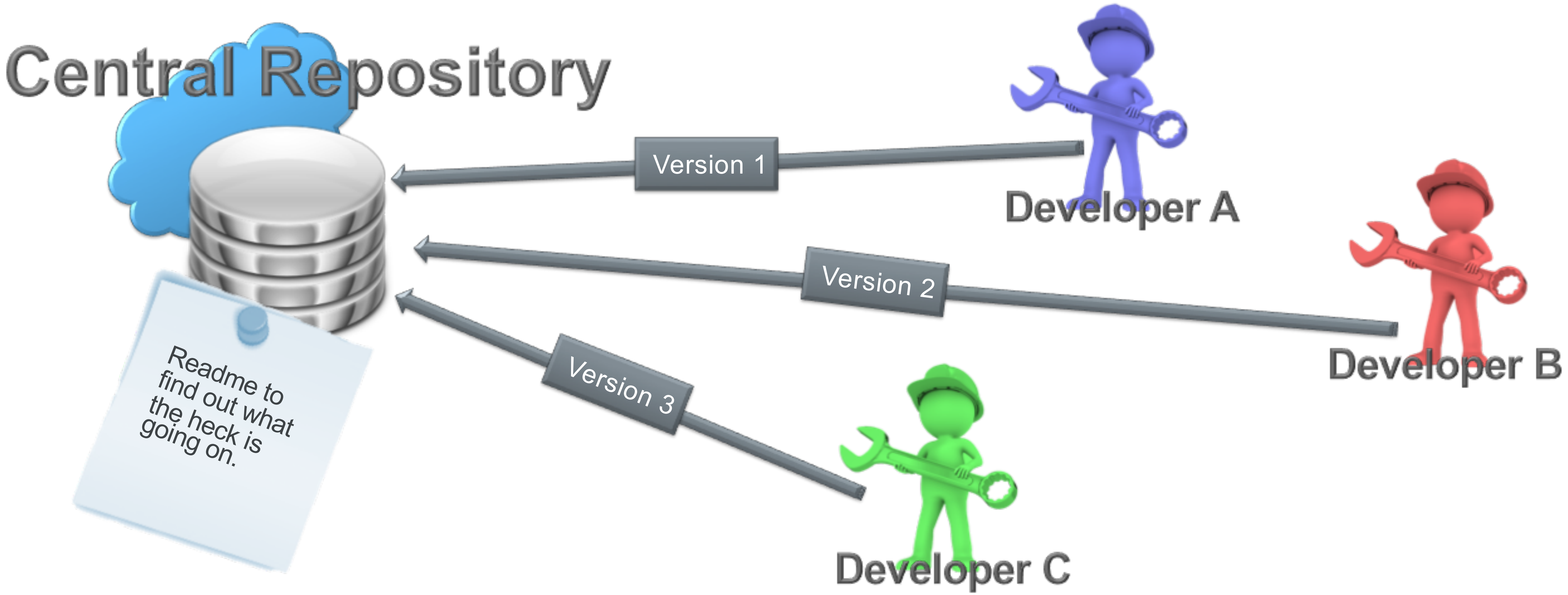
After completing this module, you should be able to:

- ➢ Generate a Chef cookbook
- ➢ Define a Chef recipe that sets up a web server
- ➢ Use Test Kitchen to verify your recipes converge on a virtual instance
- ➢ Use Inspec to test your cookbook

Questions You May Have

1. Thinking about recipes, 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



Versioning Pros and Cons

```
$ cp setup.rb setup.rb.bak
or
$ cp setup{,.`date +%Y%m%d%H%M`}
or
$ cp setup{,.`date +%Y%m%d%H%M`-`$USER`}
```

Saving a copy of the original file as another filename.

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.

GL: Create a Cookbook

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

Objective:

- ❑ Use chef to generate a 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
```

CONCEPT

What is 'chef'?

An executable program that allows you generate cookbooks and cookbook components.

CHEF

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/webserver
```

```
Generating cookbook webserver
- 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/webserver` 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 /f cookbooks\webserver
```

```
webserver
├── Berksfile
├── 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 /f cookbooks\webserver
```

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

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

```
> gc cookbooks\webserver\metadata.rb
```

```
name              'webserver'
maintainer        'The Authors'
maintainer_email  'you@example.com'
license           'all_rights'
description       'Installs/Configures webserver'
long_description  'Installs/Configures webserver'
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>/webserver/issues' if
respond_to?(:issues_url)
```

GL: The Cookbook Has a Folder for Recipes

```
> tree /f cookbooks\webserver
```

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

CHEF

GL: The Cookbook Has a Default Recipe

```
> gc cookbooks\webserver\recipes\default.rb
```

```
#
# Cookbook:: webserver
# Recipe:: default
#
# Copyright:: 2017, The Authors, All Rights Reserved.
```

GL: Create a Cookbook

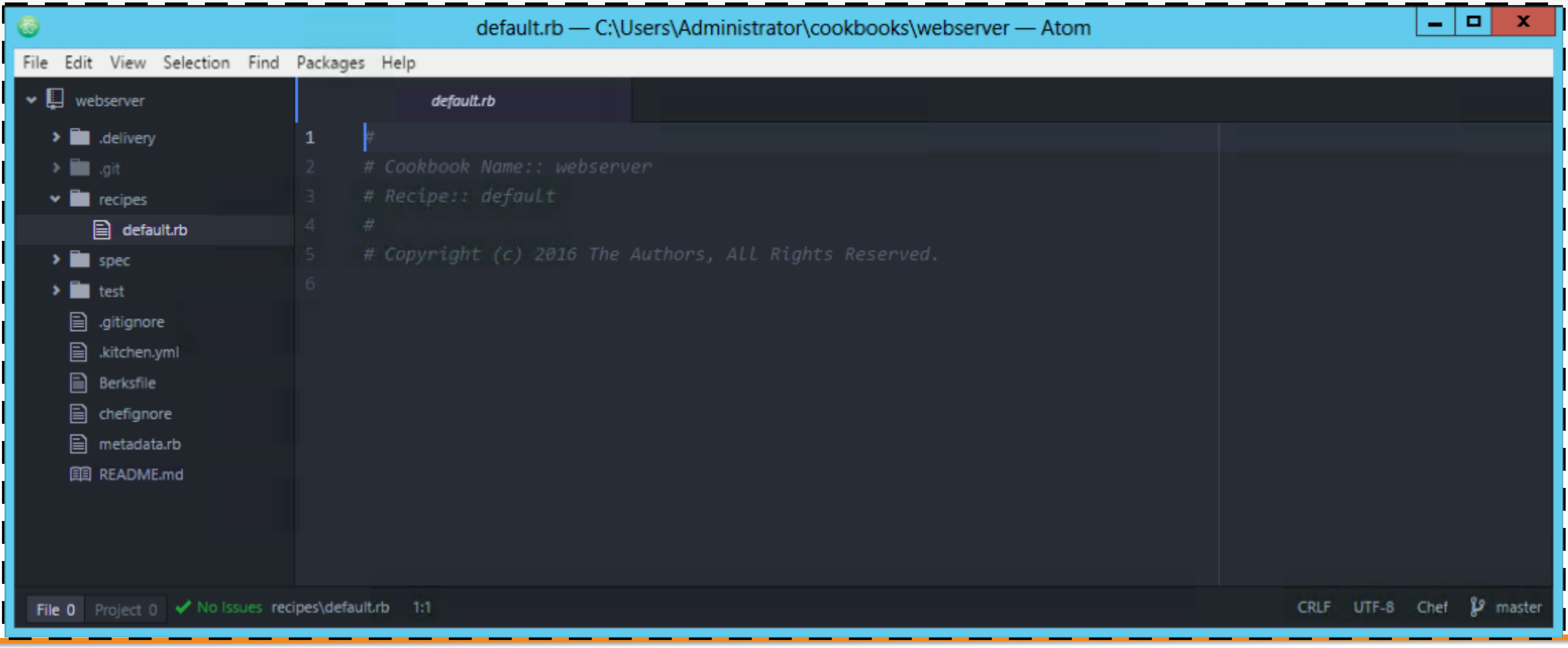
How are we going to manage this file? Does it need a README?

Objective:

- ✓ Use chef to generate a cookbook
- ❑ Add the new cookbook to version control

GL: Use Your Editor to Open the Cookbook

```
> code cookbooks\webserver
```


default.rb — C:\Users\Administrator\cookbooks\webserver — Atom
File Edit View Selection Find Packages Help
webserver
.delivery
.git
recipes
default.rb
spec
test
.gitignore
.kitchen.yml
Berksfile
chefignore
metadata.rb
README.md
default.rb
File 0 Project 0 No Issues recipes\default.rb 1:1
CRLF UTF-8 Chef master

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

GL: Update the Default Recipe

```
cookbooks\webserver\recipes\default.rb
```

```
package 'httpd' do
  action :install
end

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

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

GL: Move into the Cookbook Directory

```
> cd cookbooks\webserver
```

REMOTE

GL: Initialize the Directory as a git Repository

```
> git init
```

```
Reinitialized existing Git repository in /home/chef/cookbooks/webserver/.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

CHEF

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

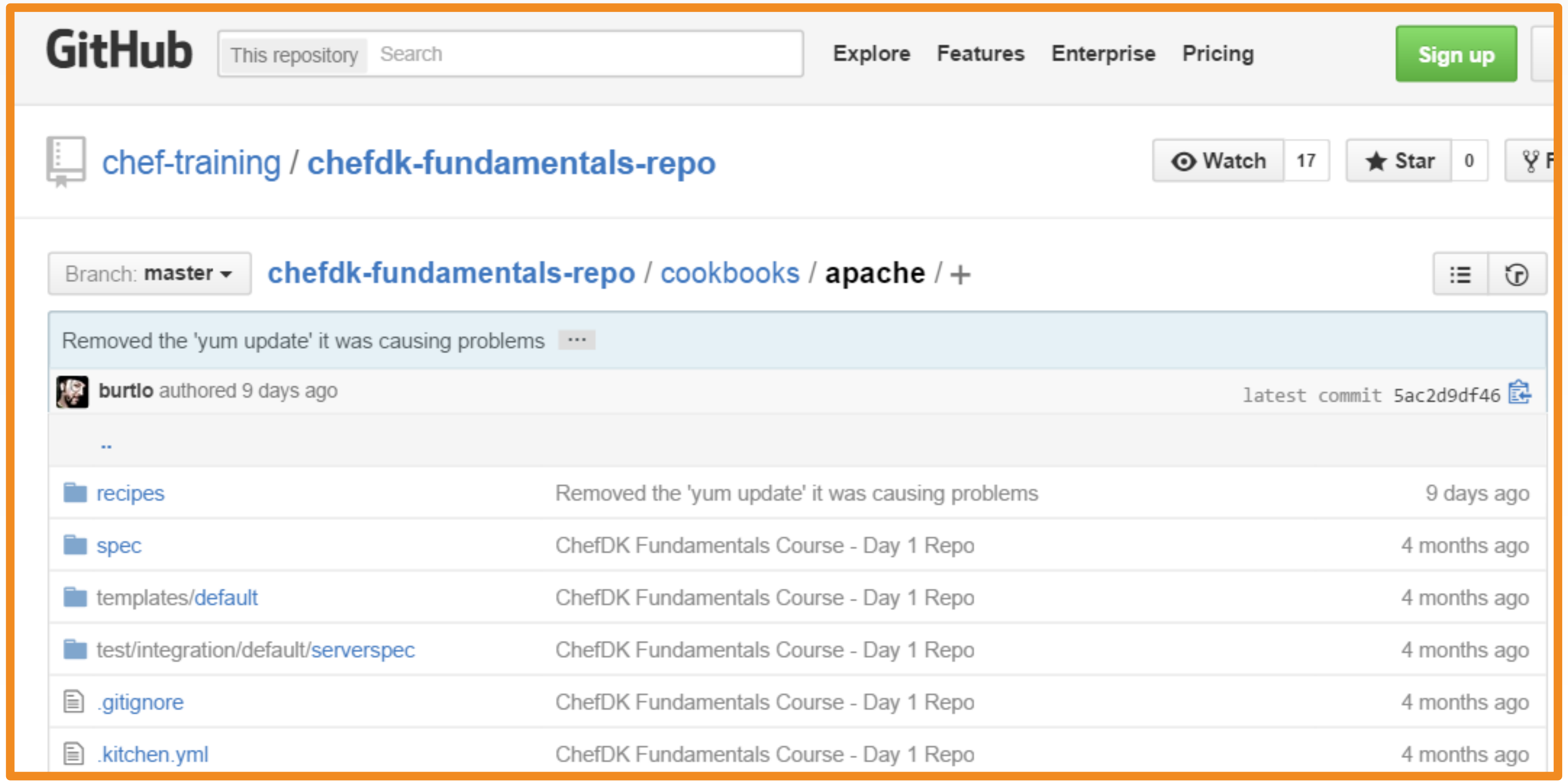
```
> git commit -m "Initial commit"
```

```
master (root-commit) 9998472] Initial webserver cookbook
 Committer: ChefDK User <chef@ip-172-31-59-191.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:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

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 ~
```

REMOTE

Can We Test Cookbooks?

As we start to define our infrastructure as code we also need to start thinking about testing it.

Mandating Testing

What steps would it take to test one of the cookbooks that we have created?

Steps to Verify Cookbooks

Create Virtual Machine

Install Chef Tools

Copy Cookbooks

Run/Apply Cookbooks

Verify Assumptions

Destroy Virtual Machine

Testing Cookbooks

We can start by first mandating that all cookbooks are tested

How often should you test your cookbook?

How often do you think changes will occur?

What happens when the rate of cookbook changes exceed the time interval it takes to verify the cookbook?

CHEF

CONCEPT

Code Testing

An automated way to ensure code accomplishes the intended goal and help the team understand its intent

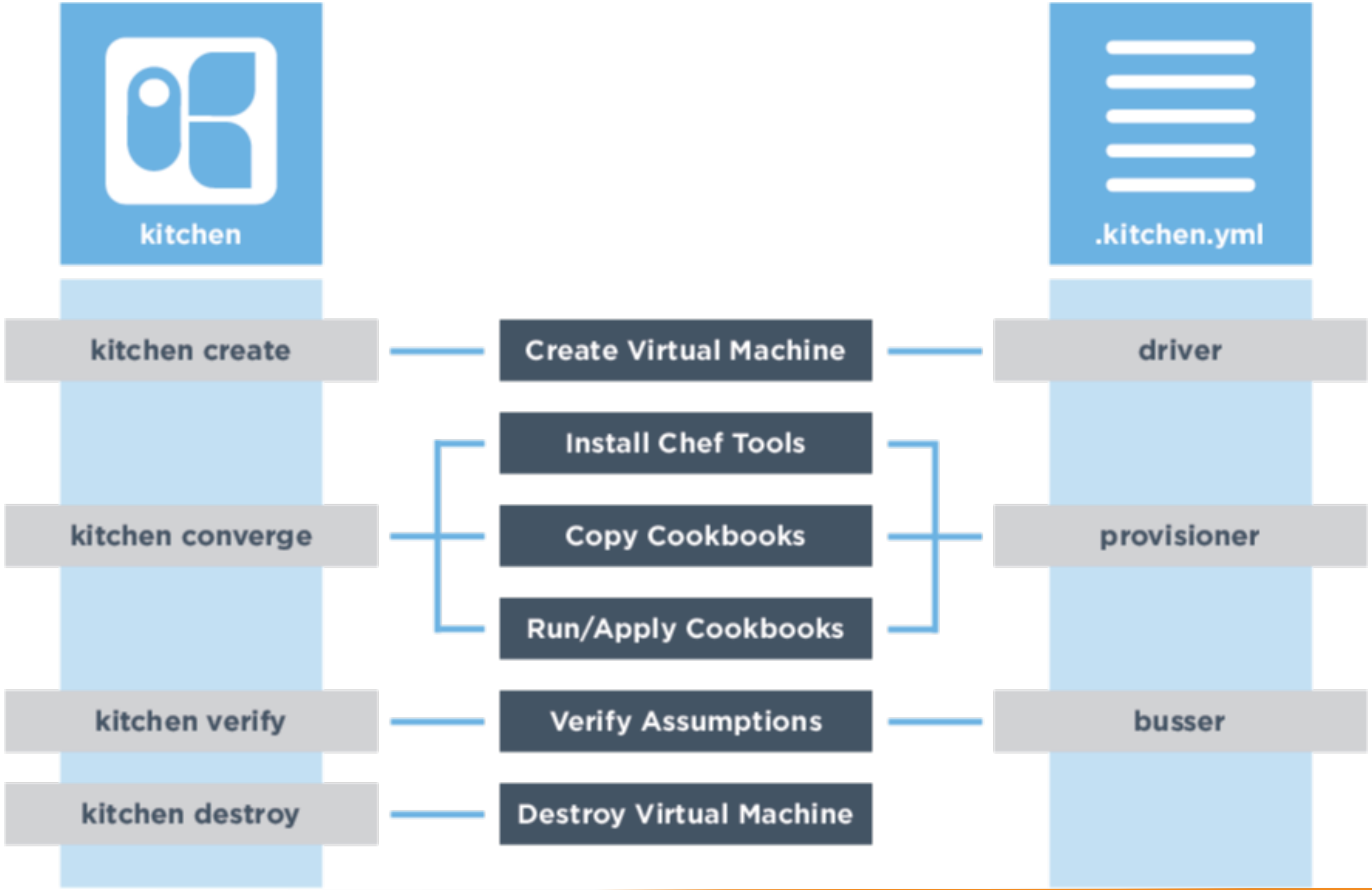
Test Configuration

What are we running in production? Maybe I could test the cookbook against a virtual machine.

Objective:

- ❑ Configure the ”webserver" cookbook to test against the centos-6.7 platform
- ❑ Test the "webserver" cookbook on a virtual machine

Test Kitchen Commands and Configuration



What Can 'kitchen' Do?

```
> kitchen --help
```

```
Commands:
  kitchen console                             # Kitchen Console!
  kitchen converge [INSTANCE|REGEXP|all]      # Converge one or more instances
  kitchen create [INSTANCE|REGEXP|all]        # Create one or more instances
  kitchen destroy [INSTANCE|REGEXP|all]       # Destroy one or more instances
  ...
  kitchen help [COMMAND]                      # Describe available commands or one specif...
  kitchen init                                # Adds some configuration to your cookbook...
  kitchen list [INSTANCE|REGEXP|all]          # Lists one or more instances
  kitchen setup [INSTANCE|REGEXP|all]         # Setup one or more instances
  kitchen test [INSTANCE|REGEXP|all]          # Test one or more instances
  kitchen verify [INSTANCE|REGEXP|all]        # Verify one or more instances
  kitchen version                             # Print Kitchen's version information
```

CHEF

Do We Have a .kitchen.yml?

```
> tree /f cookbooks\webserver
```

```
...
├── .kitchen.yml
├── metadata.rb
├── README.md
├── recipes
│   ├── default.rb
│   └── setup.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
│           └── default_spec.rb
└── test
```

What is Inside .kitchen.yml?

```
> gc cookbooks\webserver\.kitchen.yml
```

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
```

CONCEPT

.kitchen.yml

When chef generates a cookbook, a default .kitchen.yml is created. It contains kitchen configuration for the driver, provisioner, platform, and suites.

http://kitchen.ci/docs/getting-started/creating-cookbook

CHEF

Demo: The kitchen Driver

~/cookbooks/webserver/.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
```

The driver is responsible for creating a machine that we'll use to test our cookbook.

Example Drivers:

- docker
- vagrant

Demo: The kitchen Provisioner

~\cookbooks\webserver\.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
```

This tells Test Kitchen how to run Chef, to apply the code in our cookbook to the machine under test.

The default and simplest approach is to use chef_zero.

CHEF

Demo: The kitchen Verifier

~\cookbooks\webserver\.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
```

This tells Test Kitchen how to verify the converged instances.

The default approach is to use InSpec.

CHEF

Demo: The kitchen Platforms

~\cookbooks\webserver\.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
```

This is a list of operation systems on which we want to run our code.

Demo: The kitchen Suites

~\cookbooks\webserver\.kitchen.yml

```
...

suites:
  - name: default
    run_list:
      - recipe[webserver::default]
    verifier:
      inspec_tests:
        - test/recipes
    attributes:
```

This section defines what we want to test. It includes the Chef run-list of recipes that we want to test.

We define a single suite named "default".

Demo: The kitchen Suites

~\cookbooks\webserver\.kitchen.yml

```
...

suites:
  - name: default
    run_list:
      - recipe[webserver::default]
    verifier:
      inspec_tests:
        - test/smoke/default
    attributes:
```

The suite named "default" defines a run_list.

Run the ”webserver" cookbook's "default" recipe file.

CONCEPT

Kitchen Test Matrix

Kitchen defines a list of instances, or test matrix, based on the platforms multiplied by the suites.

PLATFORMS x SUITES

Running kitchen list will show that matrix.

CHEF

Example: Kitchen Test Matrix

```
> kitchen list


Instance             Driver   Provisioner  Verifier  Transport Last Action
default-ubuntu-1204  Vagrant  ChefZero     Busser    Ssh       <Not Created>
default-centos-65    Vagrant  ChefZero     Busser    Ssh       <Not Created>
```

```
suites:
  - name: default
    run_list:
      - recipe[webserver::default]
    attributes:
```

```
platforms:
  - name: ubuntu-12.04
  - name: centos-6.5
```

Example: Kitchen Test Matrix

```
> kitchen list

Instance             Driver   Provisioner  Verifier  Transport Last Action
default-ubuntu-1204  Vagrant  ChefZero     Busser    Ssh       <Not Created>
default-centos-65    Vagrant  ChefZero     Busser    Ssh       <Not Created>
```

```
suites:
  - name: default
    run_list:
      - recipe[webserver::default]
    attributes:
```

```
platforms:
  - name: ubuntu-12.04
  - name: centos-6.5
```

Group Exercise: Test Configuration

What are we running in production? Maybe I could test the cookbook against a virtual machine.

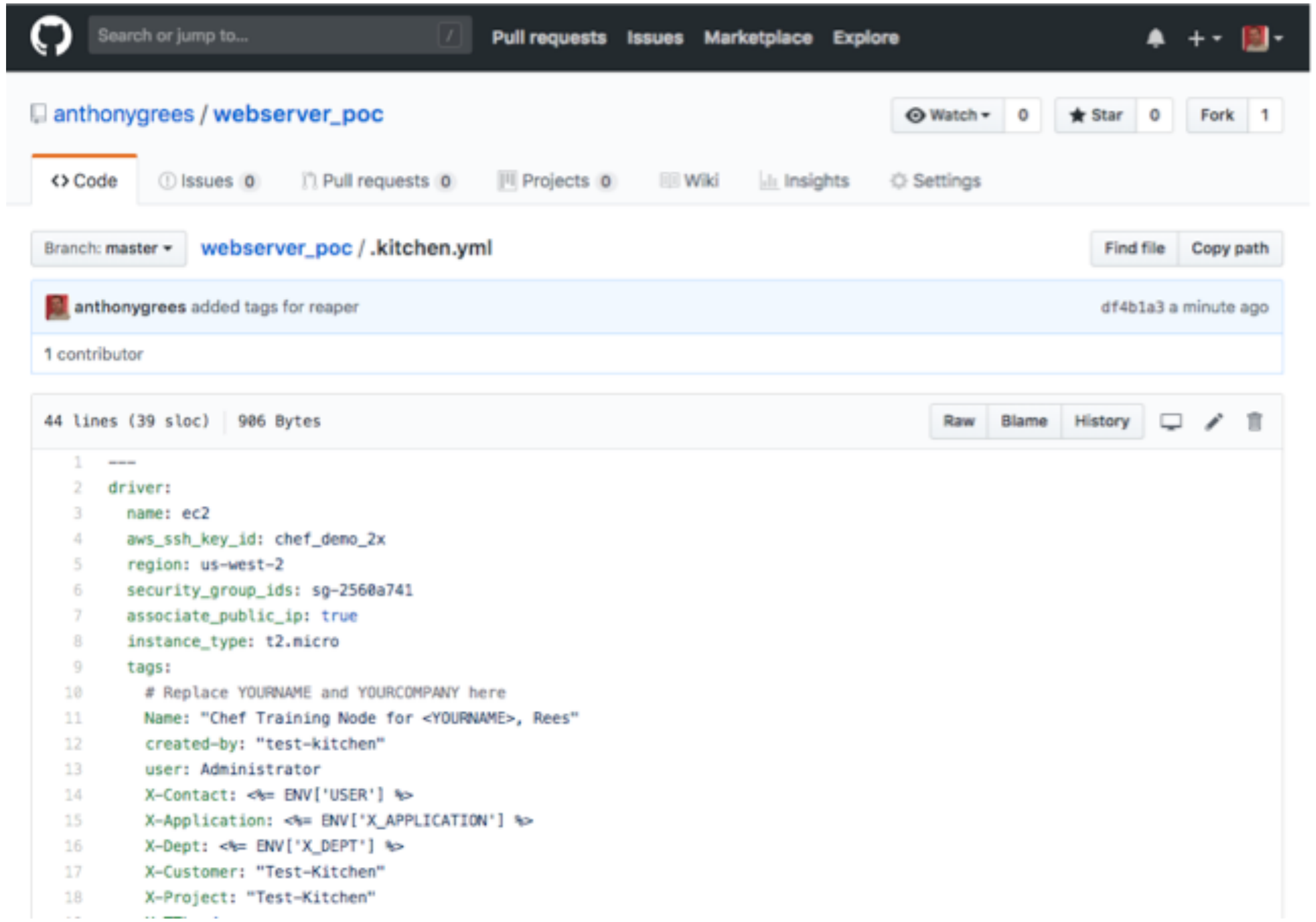
Objective:

- ❑ Configure the "webserver" cookbook's .kitchen.yml to use the EC2 driver and centos 6.x platform
- ❑ Use kitchen converge to apply the recipe on a virtual machine

GL: Move into the Cookbook's Directory

```
> cd cookbooks\webserver
```

Copy/Replace .kitchen.yml



webserver_poc / .kitchen.yml

anthonygrees added tags for reaper df4b1a3 a minute ago

1 contributor

44 lines (39 sloc) | 906 Bytes

```yaml
---
driver:
  name: ec2
  aws_ssh_key_id: chef_demo_2x
  region: us-west-2
  security_group_ids: sg-2560a741
  associate_public_ip: true
  instance_type: t2.micro
  tags:
    # Replace YOURNAME and YOURCOMPANY here
    Name: "Chef Training Node for <YOURNAME>, Rees"
    created-by: "test-kitchen"
    user: Administrator
    X-Contact: <%= ENV['USER'] %>
    X-Application: <%= ENV['X_APPLICATION'] %>
    X-Dept: <%= ENV['X_DEPT'] %>
    X-Customer: "Test-Kitchen"
    X-Project: "Test-Kitchen"
```

https://bit.ly/2J3mTIC

GL: Look at the Test Matrix

```
> kitchen list
```

```
Instance            Driver   Provisioner   Verifier   Transport   Last Action
default-centos-6    Ec2      ChefZero      Inspec     Ssh         <Not Created>
```

Converging a Cookbook

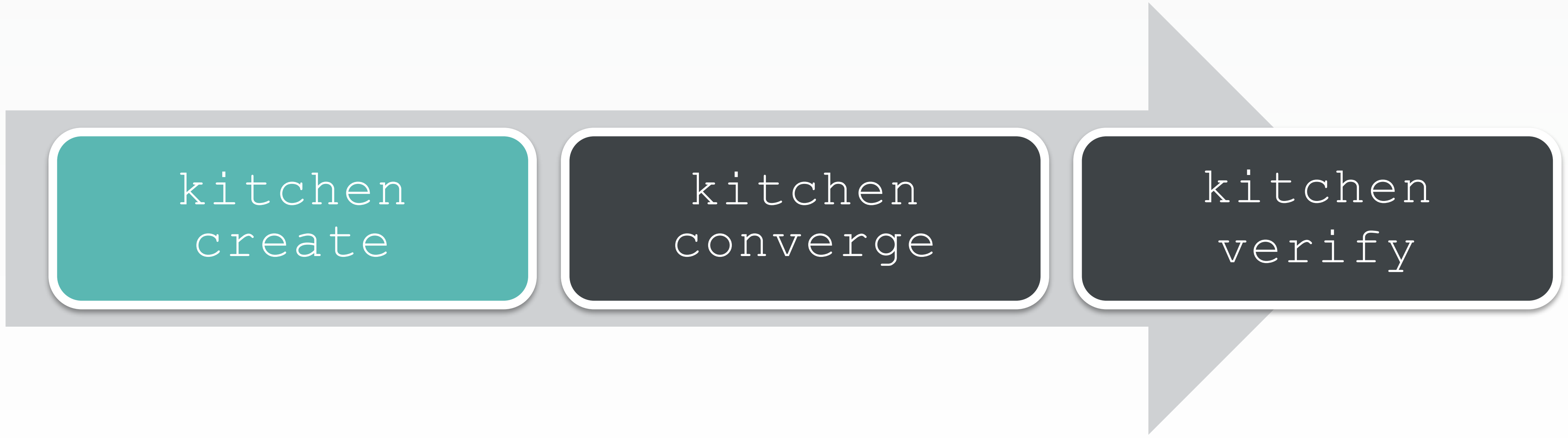
Before I add features it really would be nice to test these cookbooks against the environments that resemble production.

Objective:

✓ Configure the "webserver" cookbook's .kitchen.yml to use the ec2 driver and centos-6.x platform

❑ Use kitchen converge to apply the recipe on a virtual machine

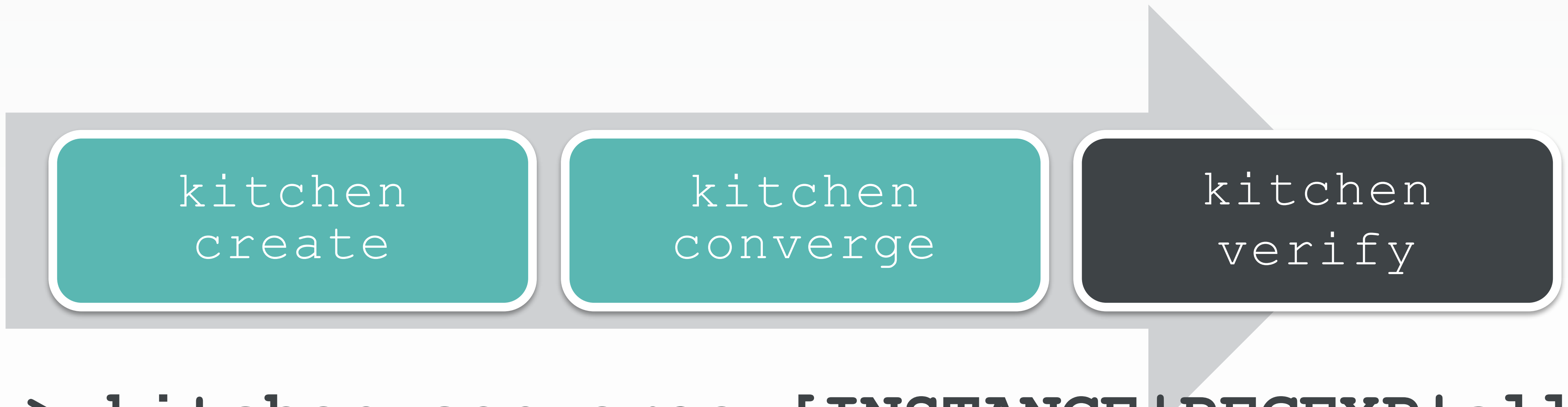
Kitchen Create



```
> kitchen create [INSTANCE|REGEXP|all]
```

Create one or more instances.

Group Exercise: Kitchen Converge



```
> kitchen converge [INSTANCE|REGEXP|all]
```

Create the instance (if necessary) and then apply the run list to one or more instances.

GL: Converge the Cookbook

```
> kitchen converge
```

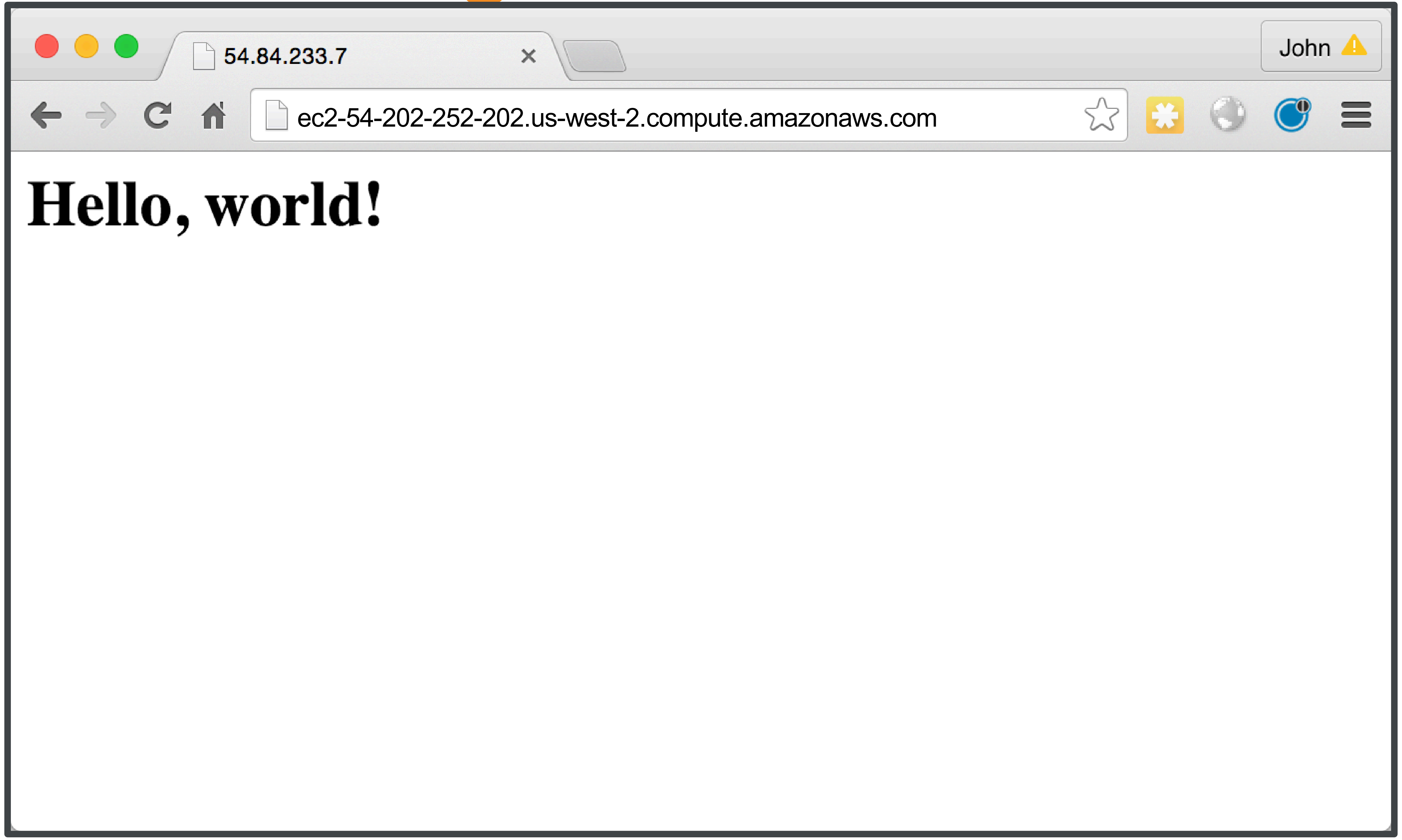
```
-----> Starting Kitchen (v1.13.2)
-----> Creating <default-centos-6>...
       Detected platform: centos version 6 on x86_64. Instance Type: t2.micro.
Default username: centos (default).
       If you are not using an account that qualifies under the AWS
free-tier, you may be charged to run these suites. The charge
should be minimal, but neither Test Kitchen nor its maintainers
are responsible for your incurred costs.

       Instance <i-b2bb691c> requested.
       Polling AWS for existence, attempt 0...
       Attempting to tag the instance, 0 retries
       EC2 instance <i-b2bb691c> created.
```

GL: Retrieve the FQDN for your instance

```
> gc .kitchen\default-centos-6.yml
```

```
---
username: centos
server_id: i-d63ba942
hostname: ec2-54-202-252-202.us-west-2.compute.amazonaws.com
last_action: converge
```

GL: Testing Our Websites



Hello, world!

DISCUSSION

Test Kitchen

What is being tested when kitchen converges a recipe without error?

What is NOT being tested when kitchen converges the recipe without error?

DISCUSSION

Test Kitchen

What is left to validate to ensure that the cookbook successfully applied the policy defined in the recipe?

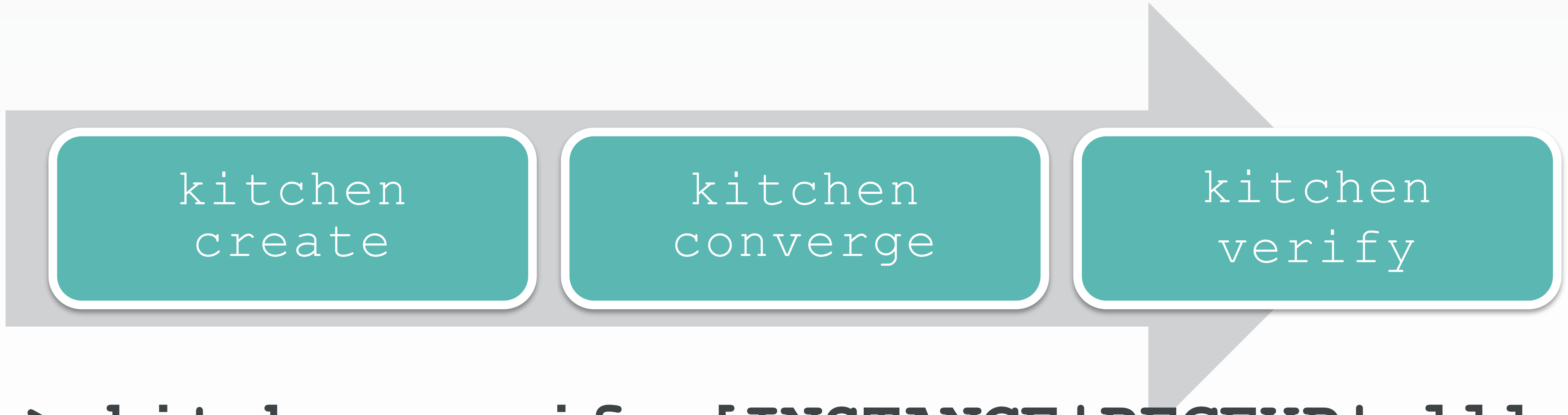
The First Test

Converging seems to validate that the recipe runs successfully. But does it assert what actually is installed?

Objective:

- ❑ In a few minutes we'll write and execute a test that asserts that the httpd package is installed when the "webserver" cookbook's default recipe is applied.

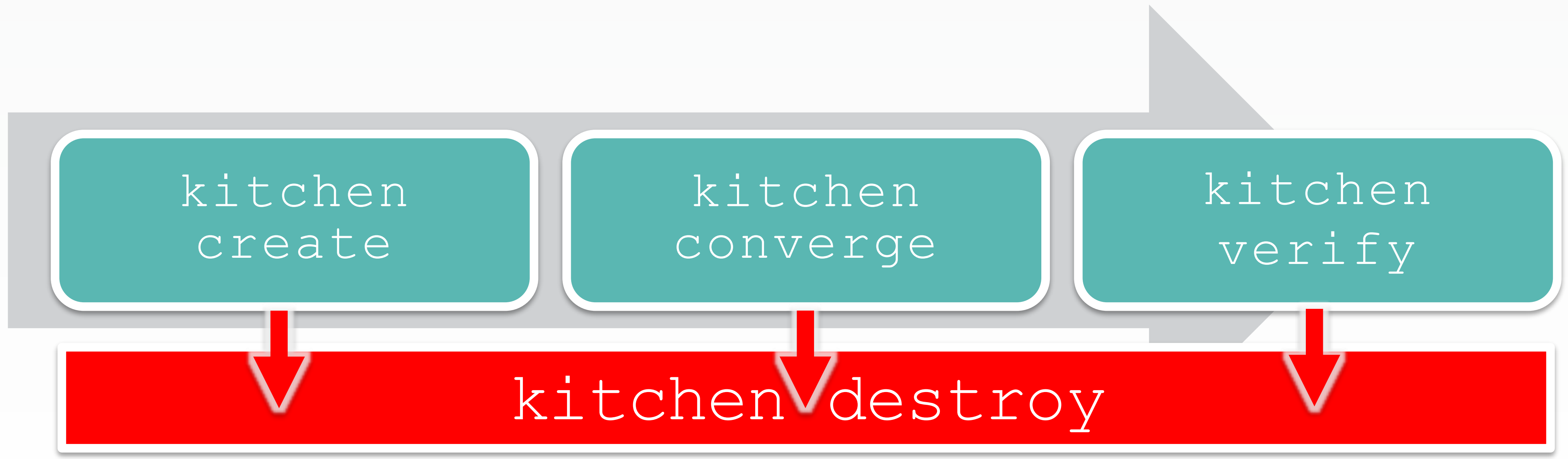
Kitchen Verify



```
> kitchen verify [INSTANCE|REGEXP|all]
```

Create, converge, and verify one or more instances.

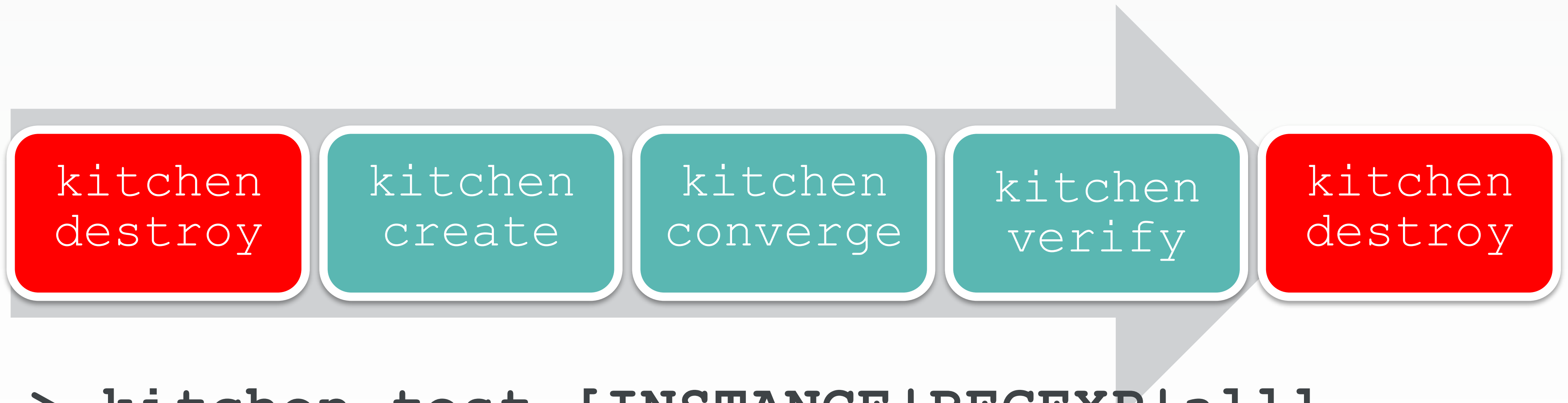
Kitchen Destroy



```
> kitchen destroy [INSTANCE|REGEXP|all]
```

Destroys one or more instances.

Kitchen Test



```
> kitchen test [INSTANCE|REGEXP|all]
```

Destroys (for clean-up), creates, converges, verifies and then destroys one or more instances.

InSpec

InSpec tests your servers' actual state by executing command locally, via SSH, via WinRM, via Docker API and so on.

https://inpsec.io

Example: Is the 'tree' Package Installed?

```
describe package('tree') do
  it { should be_installed }
end
```

I expect the package tree should be installed.

https://docs.chef.io/inspec_reference.html#id118

Example: Is the Port 80?

```
describe port(80) do
  it { should be_listening }
end
```

I expect port 80 to be listening.

https://docs.chef.io/inspec_reference.html#id118

Testing Our Webserver

I would love to know that the webserver is installed and running correctly.

Objective:

- ❑ Discuss and decide what should be tested with the webserver cookbook

DISCUSSION

Testing

What are some things we could test to validate our web server has deployed correctly?

What manual tests do we use now to validate a working web server?

Lab: Testing Webserver

- ❑ Update the test file for the ”webserver" cookbook's default recipe
- ❑ Add tests that validate a working web server

  https://www.inspec.io/docs/reference/resources/port

  https://www.inspec.io/docs/reference/resources/command
- ❑ Run kitchen verify
- ❑ Commit your changes

Lab: Return Home and 'cd cookbooks/webserver'

```
> cd ~\cookbooks\webserver
```

LOCAL

CHEF

Lab: What Does the Webserver Say?

```
~\cookbooks\webserver\test\smoke\default\default_test.rb
```

```
unless os.windows?
  describe user('root') do
    it { should exist }, :skip do
  end
end


describe port(80) do
  it { should be_listening }, :skip do
end
```

Lab: What Does the Webserver Say?

```
~\cookbooks\webserver\test\smoke\default\default_test.rb
```

```
describe port(80) do
  it { should be_listening }
end

describe command('curl localhost') do
  its('stdout') { should match('Hello, world') }
end
```

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

```
> git add .
```

REMOTE

CHEF

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

```
> git status
```

```
warning: LF will be replaced by CRLF in .kitchen.yml.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in .kitchen.yml.
The file will have its original line endings in your working directory.
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   .kitchen.yml
        modified:   test/recipes/default_test.rb
```

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

```
> git commit -m "Created initial tests"
```

```
master (root-commit) 9998472] Initial webserver cookbook
 Committer: ChefDK User <chef@ip-172-31-59-191.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:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

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

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

Lab: Verifying the Expectations

```
> kitchen verify
```

```
-----> Starting Kitchen (v1.11.1)
-----> Verifying <default-centos-67>...
       Use `/home/chef/cookbooks/webserver/test/recipes/default` for testing


Target:  ssh://kitchen@localhost:32769


  ✓  User root should exist
  ✓  Port 80 should be listening
  ✓  Command curl localhost stdout should match "Hello, world"


Summary: 3 successful, 0 failures, 0 skipped
       Finished verifying <default-centos-67> (0m0.87s).
```

DISCUSSION

Discussion

Why do you have to run kitchen within the directory of the cookbook?

Where would you define additional platforms?

Why would you define a new test suite?

What are the limitations of using Test Kitchen to validate recipes?

CHEF

DISCUSSION

Q&A

What questions can we help you answer?

- Test Kitchen
- kitchen commands
- kitchen configuration
- InSpec

CHEF™