Testing While Refactoring to Multiple Platforms

Objectives

After completing this module, you should be able to:

- ➢ Define expectations for multiple platforms
- ➢ Implement a cookbook that supports multiple platforms

EXERCISE

Node Platform in ChefSpec

What platform is the node when running a ChefSpec test?

How might you find out what is the platform?

Objective:

- ❑ Insert a break point, execute the tests, and determine the node's platform
- ❑ Remove the break point and transcend documentation

Add a Break Point to the Default Recipe

~/apache/recipes/default.rb

```
#
# Cookbook Name:: apache
# Recipe:: default
#
# Copyright (c) 2017 The Authors, All Rights Reserved.
require 'pry'                                                    +
binding.pry
include_recipe 'apache::install'
include_recipe 'apache::configuration'
include_recipe 'apache::service'
```

Execute the Tests to Initiate Pry

```
> chef exec rspec
```

```
From: /tmp/chefspec20180313-24027-
408ikafile_cache_path/cookbooks/apache/recipes/default.rb @ line 8
Chef::Mixin::FromFile#from_file:

     3: # Recipe:: default
     4: #
     5: # Copyright:: 2017, The Authors, All Rights Reserved.
     6: require 'pry'
     7: binding.pry
 =>  8: include_recipe 'apache::install'
     9: include_recipe 'apache::configuration'
```

Query the Node Object's Platform

```
[1] pry(#<Chef::Recipe>)> node['platform']
```

```
=> "centos"
```

REFERENCE

Fauxhai

ChefSpec uses the platform you specify in the runner. You can specify any platform from the list of platforms that are stored in a gem named 'Fauxhai'.

The gem contains static node objects for most major platforms and versions.

https://github.com/customink/fauxhai/tree/master/lib/fauxhai/platforms

https://github.com/customink/fauxhai

Immediately Exit the Execution

```
[2] pry(#<Chef::Recipe>)> exit!
```

EXERCISE

Node Platform in ChefSpec

What platform is the node when running a ChefSpec test?

How might you find out what is the platform?

Objective:

✓ Insert a break point, execute the tests, and determine the node's platform

❑ Remove the break point and transcend documentation

Remove the Break Point from the Recipe

~/apache/recipes/default.rb

```
#
# Cookbook Name:: apache
# Recipe:: default
#
# Copyright (c) 2017 The Authors, All Rights Reserved.
require 'pry'
binding.pry
include_recipe 'apache::install'
include_recipe 'apache::configuration'
include_recipe 'apache::service'
```

EXERCISE

Node Platform in ChefSpec

What platform is the node when running a ChefSpec test?

How might you find out what is the platform?

Objective:

- ✓ Insert a break point, execute the tests, and determine the node's platform
- ✓ Remove the break point and transcend documentation

EXERCISE

Support for CentOS & Ubuntu

The best of both worlds!

Objective:

- ❑ Write a test that verifies the Install recipe chooses the correct package on CentOS & Ubuntu
- ❑ Execute the tests and verify the tests fail
- ❑ Update the attribute to provide support for CentOS & Ubuntu
- ❑ Execute the tests and verify the tests pass

Add a Second Context for Another Platform

~/spec/unit/recipes/install_spec.rb

```
  # ... REST OF SPEC FILE ...
  context 'When all attributes are default, on Ubuntu 14.04' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu', version: '14.04')
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      expect { chef_run }.to_not raise_error
    end

    it 'installs the necessary package' do
      expect(chef_run).to install_package('apache2')
    end
  end
end
```

EXERCISE

Support for CentOS & Ubuntu

Seems like a lot of duplication but its worth it for the test coverage.

Objective:

✓ Write a test that verifies the Install recipe chooses the correct package on CentOS & Ubuntu
❑ Execute the tests and verify the tests fail
❑ Update the attribute to provide support for CentOS & Ubuntu
❑ Execute the tests and verify the tests pass

Execute the Tests to See it Fail

```
> chef exec rspec spec/unit/recipes/install_spec.rb
```

```
...F

Failures:

  1) apache::install When all attributes are default, on Ubuntu 14.04 installs
the necessary package
     Failure/Error: expect(chef_run).to install_package('apache2')

       expected "package[apache2]" with action :install to be in Chef run.
Other package resources:

         apt_package[httpd]
```

EXERCISE

Support for CentOS & Ubuntu

Failure means we have work to do!

Objective:

- ✓ Write a test that verifies the Install recipe chooses the correct package on CentOS & Ubuntu
- ✓ Execute the tests and verify the tests fail
- ❑ Update the attribute to provide support for CentOS & Ubuntu
- ❑ Execute the tests and verify the tests pass

CONCEPT

Switching on Node Platform

To control the flow of execution we need to employ some Ruby conditional statements. Conditional statements allow us to alter this control flow. Because we have access to the power of Ruby we have many choices.

https://docs.chef.io/dsl_recipe.html#sts=case Statements%C2%B6

Update the Attributes to Support Platforms

~/apache/attributes/default.rb

```
case node['platform']
when 'ubuntu'
  default['apache']['package_name'] = 'apache2'
else
  default['apache']['package_name'] = 'httpd'
end

default['apache']['package_name'] = 'httpd'
default['apache']['service_name'] = 'httpd'
default['apache']['default_index_html'] ='/var/www/html/index.html'
```

EXERCISE

Support for CentOS & Ubuntu

This should do it!

Objective:

- ✓ Write a test that verifies the Install recipe chooses the correct package on CentOS & Ubuntu
- ✓ Execute the tests and verify the tests fail
- ✓ Update the attribute to provide support for CentOS & Ubuntu
- ❑ Execute the tests and verify the tests pass

Execute the Tests to See it Pass

```
> chef exec rspec spec/unit/recipes/install_spec.rb
```

```
....

Finished in 1.35 seconds (files took 4.51 seconds to load)
4 examples, 0 failures
```

EXERCISE

Support for CentOS & Ubuntu

Woot! Multi-platform support for the installation!

Objective:

- ✓ Write a test that verifies the Install recipe chooses the correct package on CentOS & Ubuntu
- ✓ Execute the tests and verify the tests fail
- ✓ Update the attribute to provide support for CentOS & Ubuntu
- ✓ Execute the tests and verify the tests pass

CHEF

LAB

Support for CentOS & Ubuntu

- ❑ Write a test that verifies the Service recipe chooses the service named 'httpd' on CentOS and 'apache2' on Ubuntu
- ❑ Execute the tests and verify the tests **fail**
- ❑ Update the attribute to choose the service name 'httpd' on CentOS and 'apache2' on Ubuntu
- ❑ Execute the tests and verify the tests **pass**

Add a Second Context for Another Platform

~/spec/unit/recipes/service_spec.rb

```
  # ... REST OF SPEC FILE ...
  context 'When all attributes are default, on Ubuntu 14.04' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu', version: '14.04')
      runner.converge(described_recipe)
    end
    # ... it converges successfully ...

    it 'starts the appropriate service' do
      expect(chef_run).to start_service('apache2')
    end
    it 'enables the appropriate service' do
      expect(chef_run).to enable_service('apache2')
    end
  end
end
```

Execute the Tests to See it Fail

```
> chef exec rspec spec/unit/recipes/service_spec.rb
```

```
....FF

Failures:

  1) apache::service When all attributes are default, on an Ubuntu 14.04
starts the necessary service
     Failure/Error: expect(chef_run).to start_service('apache2')

       expected "service[apache2]" with action :start to be in Chef run. Other
service resources:

         service[httpd]
```

Update the Attribute to Support Platforms

~/apache/attributes/default.rb

```
case node['platform']
when 'ubuntu'
  default['apache']['package_name'] = 'apache2'
  default['apache']['service_name'] = 'apache2'
else
  default['apache']['package_name'] = 'httpd'
  default['apache']['service_name'] = 'httpd'
end

default['apache']['service_name'] = 'httpd'
default['apache']['default_index_html'] = '/var/www/html/index.html'
```

Execute the Tests to See it Pass

```
> chef exec rspec spec/unit/recipes/service_spec.rb
```

```
......

Finished in 1.84 seconds (files took 4.22 seconds to load)
6 examples, 0 failures
```

LAB

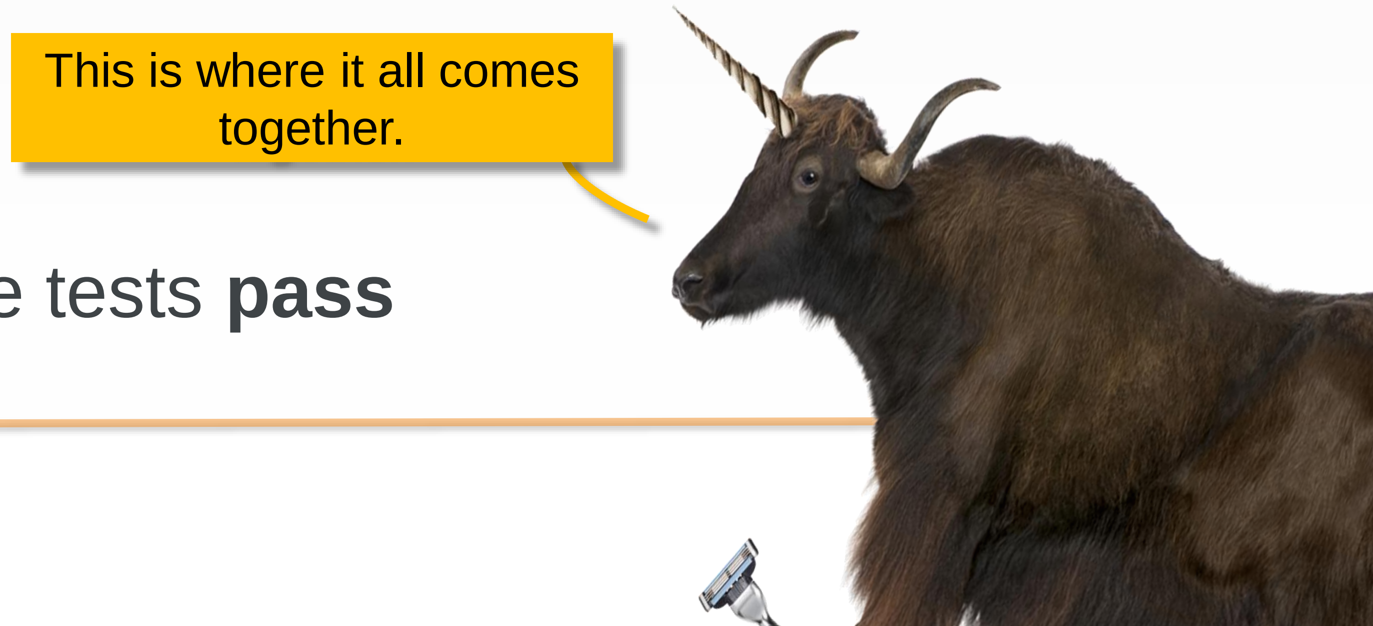
Support for CentOS & Ubuntu

- ✓ Write a test that verifies the Service recipe chooses the service named 'httpd' on CentOS and 'apache2' on Ubuntu
- ✓ Execute the tests and verify the tests **fail**
- ✓ Update the attribute to choose the service name 'httpd' on CentOS and 'apache2' on Ubuntu
- ✓ Execute the tests and verify the tests **pass**

LAB

Support for CentOS & Ubuntu

- ❑ Write a test that verifies the file recipe chooses the same path (name) '/var/www/html/index.html' on CentOS and on Ubuntu
- ❑ Execute the tests that verify the tests **pass**
- ❑ Update the attribute to choose the same path on CentOS and on Ubuntu
- ❑ Execute the tests that verify the tests **pass**
- ❑ Get nervous! Mutate the attributes file!
- ❑ Undo the entire attributes change and verify the tests **pass**



Add a Second Context for Another Platform

~/spec/unit/recipes/configuration_spec.rb

```
  # ... REST OF SPEC FILE ...
  context 'When all attributes are default, on Ubuntu 14.04' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu',version: '14.04')
      runner.converge(described_recipe)
    end
    # ... it converges successfully ...

 it 'creates the index.html' do
      expect(chef_run).to render_file('/var/www/html/index.html').with_content('<h1>Welcome
Home!</h1>')
    end
  end
end
```

Execute the Tests to See it Pass

```
> chef exec rspec spec/unit/recipes/configuration_spec.rb
```

```
....


Finished in 1.84 seconds (files took 4.22 seconds to load)
4 examples, 0 failures
```

Update the Attribute to Support Platforms

~/apache/attributes/default.rb

```
case node['platform']
when 'ubuntu'
  default['apache']['package_name'] = 'apache2'
  default['apache']['service_name'] = 'apache2'
  default['apache']['default_index_html'] = '/var/www/html/index.html'
else
  default['apache']['package_name'] = 'httpd'
  default['apache']['service_name'] = 'httpd'
  default['apache']['default_index_html'] = '/var/www/html/index.html'
end

default['apache']['default_index_html'] = '/var/www/html/index.html'
```

Execute the Tests to See it Pass

```
> chef exec rspec spec/unit/recipes/configuration_spec.rb
```

```
....


Finished in 1.84 seconds (files took 4.22 seconds to load)
4 examples, 0 failures
```

Heckle the code

```
~/apache/attributes/default.rb
```

```
case node['platform']
when 'ubuntu'
  default['apache']['package_name'] = 'apache2'
  default['apache']['service_name'] = 'apache2'
  default['apache']['default_index_html'] = '/var/www/html/index.html2'
else
  default['apache']['package_name'] = 'httpd'
  default['apache']['service_name'] = 'httpd'
  default['apache']['default_index_html'] = '/var/www/html/index.html'
end
```

Execute the Tests to See it Pass

```
> chef exec rspec spec/unit/recipes/configuration_spec.rb
```

```
...F

Failures:

  1) apache::configuration When all attributes are default, on an Ubuntu 14.04 creates the index.html
     Failure/Error: expect(chef_run).to render_file('/var/www/html/index.html').with_content('<h1>Welcome Home!</h1>')

       expected Chef run to render "/var/www/html/index.html"
```

Update the Attributes

~/apache/attributes/default.rb

```
case node['platform']
when 'ubuntu'
  default['apache']['package_name'] = 'apache2'
  default['apache']['service_name'] = 'apache2'
  default['apache']['default_index_html'] = '/var/www/html/index.html2'
else
  default['apache']['package_name'] = 'httpd'
  default['apache']['service_name'] = 'httpd'
  default['apache']['default_index_html'] = '/var/www/html/index.html'
end

default['apache']['default_index_html'] = '/var/www/html/index.html'
```

Execute the Tests to See them Pass

```
> chef exec rspec
```

```
..................


Finished in 6.02 seconds (files took 4.02 seconds to load)
18 examples, 0 failures
```

LAB

Support for CentOS & Ubuntu

- ✓ Write a test that verifies the file recipe chooses the same path (name) '/var/www/html/index.html' on CentOS and on Ubuntu
- ✓ Execute the tests that verify the tests **pass**
- ✓ Update the attribute to choose the same path on CentOS and on Ubuntu
- ✓ Execute the tests that verify the tests **pass**
- ✓ Get nervous! Mutate the attributes file!
- ✓ Undo the entire attributes change and verify the tests **pass**

PROBLEM

What About an Integration Test

Remember that ChefSpec and Fauxhai are fake in-memory representations of a chef-client run. They are not equivalent to running the recipe on the specified platform.

CHEF

EXERCISE

Integration Test with Ubuntu

This is where it all started.

Objective:

- ❑ Update the Kitchen Configuration to test on Ubuntu
- ❑ Execute the integration tests and verify that they pass

CHEF

Add a New Platform to the Kitchen Configuration

~/apache/.kitchen.yml

```
---
driver:
  name: docker

provisioner:
  name: chef_zero

verifier:
  name: inspec

platforms:
  - name: centos-6.9
  - name: ubuntu-14.04
```

Verify the New Instance is Present

```
> kitchen list
```

```
Instance              Driver  Provisioner  Verifier  Transport  Last Action
default-centos-69     Docker  ChefZero     InSpec    Ssh        Verified
default-ubuntu-1404   Docker  ChefZero     InSpec    Ssh        <Not Created>
```

EXERCISE

Integration Test with Ubuntu

Fingers crossed

Objective:

- ✓ Update the Kitchen Configuration to test on Ubuntu
- ❑ Execute the integration tests and verify that they pass

Execute the Tests for All Platforms

```
> kitchen test
```

```
-----> Starting Kitchen (v1.19.1)
-----> Cleaning up any prior instances of <default-centos-69>
-----> Destroying <default-centos-69>...
       Finished destroying <default-centos-69> (0m0.00s).
-----> Testing <default-centos-69>
-----> Creating <default-centos-69>
       ...
       ...
```

EXERCISE

Integration Test with Ubuntu

Now I'm sure the cookbook works on two platforms and it would be easy to add a third ... or fourth.

Objective:

- ✓ Update the Kitchen Configuration to test on Ubuntu
- ✓ Execute the integration tests and verify that they pass

Discussion

What are the benefits and drawbacks of defining unit tests for multiple platforms?

What are the benefits and drawbacks of defining integration tests for multiple platforms?

When testing multiple platforms would you start with integration tests or unit tests?

DISCUSSION

Q&A

What questions can we answer for you?

You did it!

CHEF™