Testing Resources in Recipes

Objectives

After completing this module, you should be able to:

- Test resources within a recipe using ChefSpec

EXERCISE

Testing Remaining Resources

No resources left behind!

Objective:

❑ Write and execute tests for the Install recipe
❑ Verify the test validates the recipe

Generated Recipes Also Generate Specs

```
> tree spec
```

```
spec
├── spec_helper.rb
└── unit
    └── recipes
        ├── configuration_spec.rb
        ├── default_spec.rb
        ├── install_spec.rb
        └── service_spec.rb
```

Execute the Install Specification

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

```
..


Finished in 0.7663 seconds (files took 1.86 seconds to load)
2 examples, 0 failures
```

Delete context for Ubuntu

~/apache/spec/unit/recipes/install_spec.rb

```
require 'spec_helper'

describe 'apache::install' do
 context 'When all attributes are default, on Ubuntu 16.04' do
    let(:chef_run) do
      # for a complete list of available platforms and versions see:
      # https://github.com/customink/fauxhai/blob/master/PLATFORMS.md
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu', version: '16.04')
      runner.converge(described_recipe)
    end

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

Update the ChefSpec Platform

~/apache/spec/unit/recipes/install_spec.rb

```
require 'spec_helper'

describe 'apache::install' do
  context 'When all attributes are default, on an CentOS 6.9' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'centos', version: '6.9')
      runner.converge(described_recipe)
    end

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

Add a Pending Test to Verify the Package

~/apache/spec/unit/recipes/install_spec.rb

```
    # ... START OF THE SPEC FILE ...

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

    it 'installs the necessary package'
  end
end
```

REFERENCE

ChefSpec Documentation

Find within the documentation examples of testing packages

https://github.com/chefspec/chefspec/tree/master/examples/package

Write the Test to Verify the Package

~/apache/spec/unit/recipes/install_spec.rb

```
    # ... START OF THE SPEC FILE ...

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

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

Write the Test to Verify the Package

~/apache/spec/unit/recipes/install_spec.rb

```
    # ... START OF THE SPEC FILE ...

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

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

resource's name

resource's action

resource

CHEF

Execute the Test to See it Pass

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

```
..


Finished in 0.73662 seconds (files took 4.4 seconds to load)
2 examples, 0 failures
```

EXERCISE

Testing Remaining Resources

No resources left behind!

Objective:

- ✓ Write and execute tests for the Install recipe
- ❑ Verify the test validates the recipe

PROBLEM

It's Quiet. Too Quiet.

When a test passes immediately without having to write code (or if the code has already been written) it is time to be concerned. This is one of those moments we should ensure that the tests are working by mutating that code.

Comment Out the Resource

~/apache/recipes/install.rb

```
#
# Cookbook Name:: apache
# Recipe:: install
#
# Copyright (c) 2017 The Authors, All Rights Reserved.
# package 'httpd'
```

Execute the Test to See it Fail

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

```
.F


Failures:


  1) apache::install When all attributes are default, on an CentOS 6.9 installs the appropriate package
     Failure/Error: expect(chef_run).to install_package('httpd')
       expected "package[httpd]" with action :install to be in Chef run. Other package resources:
```

Uncomment Out the Resource

~/apache/recipes/install.rb

```
#
# Cookbook Name:: apache
# Recipe:: install
#
# Copyright (c) 2017 The Authors, All Rights Reserved.
package 'httpd'
```

Execute the Test to See it Pass

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

```
..


Finished in 0.73662 seconds (files took 4.4 seconds to load)
2 examples, 0 failures
```

EXERCISE

Testing Remaining Resources

No resources left behind!

Objective:

- ✓ Write and execute tests for the Install recipe
- ✓ Verify the test validates the recipe

LAB

Test the Remaining Recipes

- ❑ Write a pending example
- ❑ Find the ChefSpec implementation
- ❑ Verify that the new example passes
- ❑ Mutate the recipe to generate a failure
- ❑ Restore the code in the recipe
- ❑ Verify that all examples pass

❖ **Repeat this series of steps for the configuration recipe and service recipe**

CHEF™

Delete context for Ubuntu

~/apache/spec/unit/recipes/service_spec.rb

```
require 'spec_helper'

describe 'apache::service' do
  context 'When all attributes are default, on Ubuntu 16.04' do
    let(:chef_run) do
      # for a complete list of available platforms and versions see:
      # https://github.com/customink/fauxhai/blob/master/PLATFORMS.md
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu', version: '16.04')
      runner.converge(described_recipe)
    end

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

Update the ChefSpec Platform

~/apache/spec/unit/recipes/service_spec.rb

```
require 'spec_helper'

describe 'apache::service' do
  context 'When all attributes are default, on an CentOS 6.9' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'centos', version: '6.9')
      runner.converge(described_recipe)
    end

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

Write the Tests to Verify the Service

~/apache/spec/unit/recipes/service_spec.rb

```
 # ... START OF THE SPEC FILE ...


     it 'starts the necessary service' do
       expect(chef_run).to start_service('httpd')
     end

     it 'enables the necessary service' do
       expect(chef_run).to enable_service('httpd')
     end
   end
end
```

CHEF

Execute the Tests to See it Pass

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

```
...


Finished in 0.93685 seconds (files took 4.28 seconds to load)
3 examples, 0 failures
```

LAB

Test the Remaining Recipes

- ✓ Write a pending example
- ✓ Find the ChefSpec implementation
- ✓ Verify that the new example passes
- ✓ Mutate the recipe to generate a failure
- ✓ Restore the code in the recipe
- ✓ Verify that all examples pass

❖ **Repeat this series of steps for the configuration recipe and service recipe**

Delete context for Ubuntu

~/apache/spec/unit/recipes/configuration_spec.rb

```
require 'spec_helper'

describe 'apache::configuration' do
  context 'When all attributes are default, on Ubuntu 16.04' do
    let(:chef_run) do
      # for a complete list of available platforms and versions see:
      # https://github.com/customink/fauxhai/blob/master/PLATFORMS.md
      runner = ChefSpec::ServerRunner.new(platform: 'ubuntu', version: '16.04')
      runner.converge(described_recipe)
    end

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

Update the ChefSpec Platform

~/apache/spec/unit/recipes/configuration_spec.rb

```
require 'spec_helper'

describe 'apache::configuration' do
  context 'When all attributes are default, on an CentOS 6.9' do
    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new(platform: 'centos', version: '6.9')
      runner.converge(described_recipe)
    end

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

Write the Tests to Verify the Service

~/apache/spec/unit/recipes/configuration_spec.rb

```
 # ... START OF THE SPEC FILE ...


    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
```

CONCEPT

rspec

When you run **`rspec`** without any paths it will automatically find and execute all the *"_spec.rb"* files within the '*spec*' directory.

CHEF

Execute All the Tests in the Spec Directory

```
> chef exec rspec
```

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


Finished in 2.31 seconds (files took 1.86 seconds to load)
11 examples, 0 failures
```

DISCUSSION

Discussion

What value does it bring to validate that the resources take the appropriate action?

DISCUSSION

Q&A

What questions can we answer for you?

Morning

Introduction

Why Write Tests? Why is that Hard?

Writing a Test First

Refactoring Cookbooks with Tests

Afternoon

Faster Feedback with Unit Testing

Testing Resources in Recipes

Refactoring to Attributes

Refactoring to Multiple Platforms

CHEF™