

# Faster Feedback with Unit Testing

# PROBLEM



## Slower Feedback Cycle

The slower the feedback loop the less value it provides to you while developing your cookbooks. You are less inclined to run the test suite. Which means you will likely miss issues as they happen.

# Objectives

After completing this module, you should be able to:

- Explain the importance and limitations of unit testing
- Write and execute a unit test

# CONCEPT

## External Dependencies

The speed of the test suite is affected by the external dependency on the creation of the test instance, installing chef, and applying the run list.



Create CentOS Instance

Install Chef

Apply the Run List

Build Node (ohai)

Synchronize Cookbooks

Build Resource Collection

Converge

Execute Tests

# CONCEPT

## Build Resource Collection

The resource collection is a list of all the resources and recipes loaded across all the recipes within the run list.



Create CentOS Instance

Install Chef

Apply the Run List

Build Node (ohai)

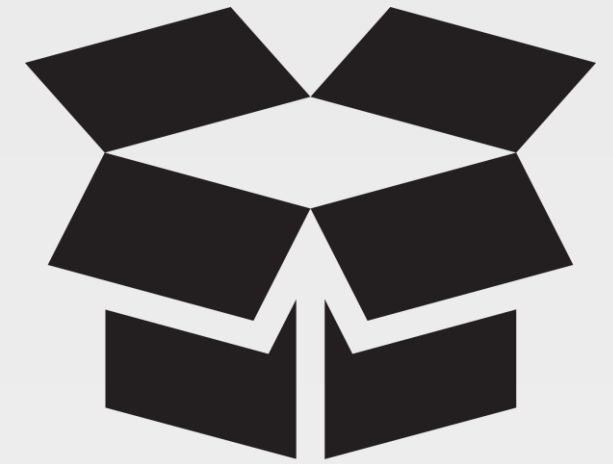
Synchronize Cookbooks

Build Resource Collection

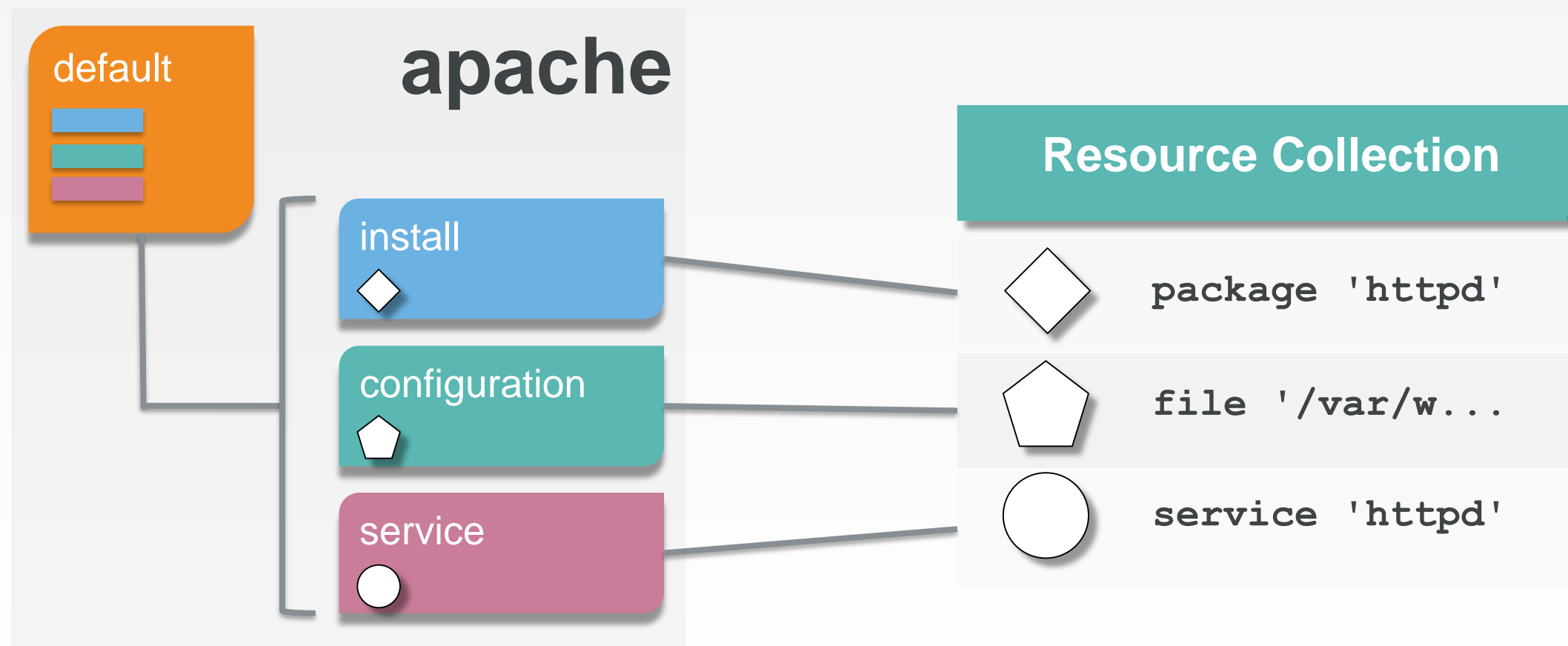
Converge

Execute Tests

# CONCEPT



## Resource Collection

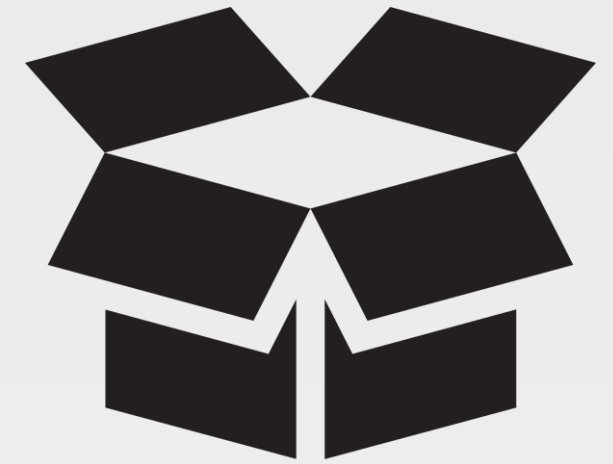


# CONCEPT

## RSpec and ChefSpec

RSpec is a Domain Specific Language (DSL) that allows you to express and execute expectations. These expectations are expressed in examples that are asserted in different example groups.

ChefSpec provides helpers and tools that allow you to express expectations about the state of **resource collection**.



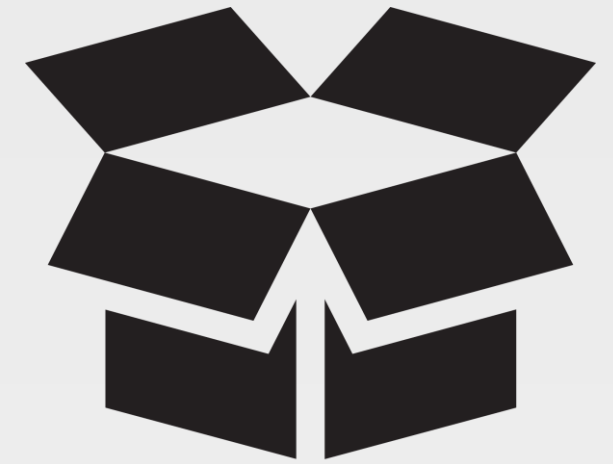
ChefSpec

RSpec

Chef

Ruby

# CONCEPT



## Test Kitchen versus ChefSpec

### ChefSpec

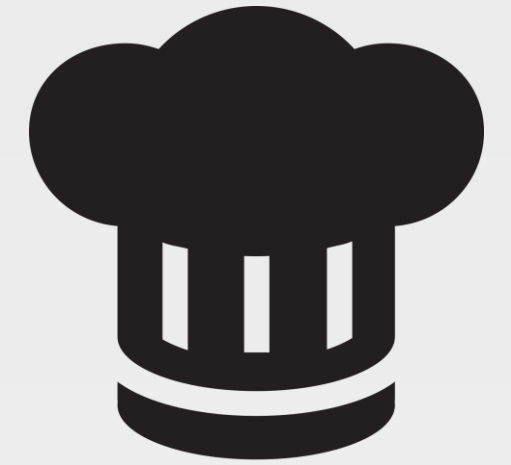


### Test Kitchen using InSpec





# EXERCISE



## Faster Feedback While Developing Cookbooks

*The faster the feedback from our tests, the more likely we are to run them.  
The more likely we are to run them means they will catch more issues.*

### Objective:

- ☐ Review and run the existing tests
- ☐ Identify the tests that we need to write
- ☐ Write and execute the tests to identify the failure
- ☐ Fix the code and execute the tests to see success

# View the Spec Directory



```
> tree spec
```

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

2 directories, 6 files
```

# View the Test for the Default Recipe

 `~/apache/spec/unit/recipes/default_spec.rb`

```
require 'spec_helper'

describe 'apache::default' do
  context 'When all attributes are default, on an Ubuntu 16.04' do
    let(:chef_run) do
      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
end
```

# View the Test for the Default Recipe

`~/apache/spec/unit/recipes/default_spec.rb`

```
require 'spec_helper'
```

example groups

```
describe 'apache::default' do
  context 'When all attributes are default, on an Ubuntu 16.04' do
    let(:chef_run) do
      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
end
```

cookbook name::recipe name

# View the Test for the Default Recipe

`~/apache/spec/unit/recipes/default_spec.rb`

```
require 'spec_helper'

describe 'apache::default' do
  context 'When all attributes are default, on an Ubuntu 16.04' do
    let(:chef_run) do
      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
end
```

example

expectation

# View the Test for the Default Recipe

`~/apache/spec/unit/recipes/default_spec.rb`

```
require 'spec_helper'

describe 'apache::default' do
  context 'When all attributes are default, on an Ubuntu 16.04' do
    let(:chef_run) do
      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
end
```

described recipe

Ruby Class

chef\_run helper

# Execute the Test for the Default Recipe



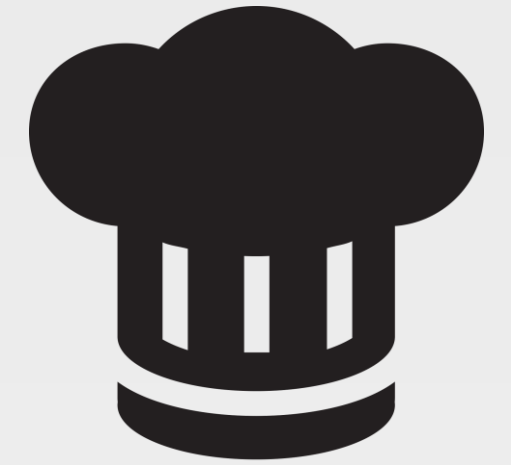
```
> chef exec rspec spec/unit/recipes/default_spec.rb
```

```
.  
Finished in 0.44592 seconds (files took 4.35 seconds to load)
```

```
1 example, 0 failures
```

passing example

# EXERCISE



## Faster Feedback While Developing Cookbooks

*The faster the feedback from our tests, the more likely we are to run them.  
The more likely we are to run them means they will catch more issues.*

### Objective:

- ✓ Review and run the existing tests
- ☐ Identify the tests that we need to write
- ☐ Write and execute the tests to identify the failure
- ☐ Fix the code and execute the tests to see success



# These are the Three Things to Test



```
~/apache/recipes/default.rb
```

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

# Update the ChefSpec Platform

```
~/apache/spec/unit/recipes/default_spec.rb
```

```
require 'spec_helper'

describe 'apache::default' 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
```

# Create a Pending Test

```
~/apache/spec/unit/recipes/default_spec.rb
```

```
# ... START OF THE SPEC FILE ...  
it 'converges successfully' do  
  expect { chef_run }.to_not raise_error  
end
```

```
it 'includes the install recipe'
```

```
end  
end
```

# Execute the Tests to See the Pending Tests



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

. \*

pending example

Pending: (Failures listed here are expected and do not affect your suite's status)

1) apache::default When all attributes are default, on an CentOS 6.9 includes the install recipe

# Not yet implemented

# ./spec/unit/recipes/default\_spec.rb:20

# ... OUTPUT CONTINUES ON NEXT SLIDE ...

# Execute the Tests to See the Pending Tests



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

```
. *
```

Pending: (Failures listed here are expected and do not affect your suite's status)

summary

1) apache::default When all attributes are default, on an CentOS 6.9 includes the install recipe

# Not yet implemented

# ./spec/unit/recipes/default\_spec.rb:20

# ... OUTPUT CONTINUES ON NEXT SLIDE ...

spec file : line number

# View the Results to See the Pending Tests



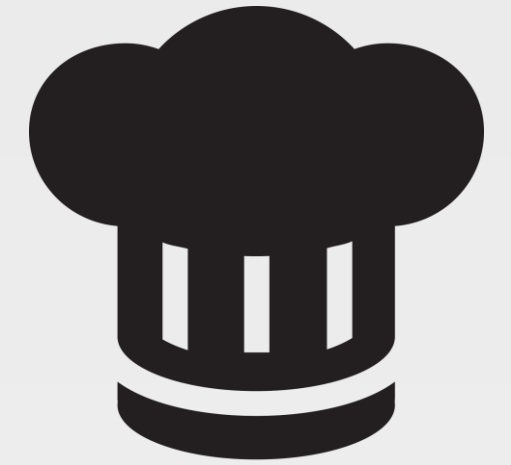
```
> chef exec rspec spec/unit/recipes/default_spec.rb
```

```
# ... OUTPUT CONTINUED FROM PREVIOUS SLIDE ...
```

```
Finished in 0.46457 seconds (files took 4.39 seconds to load)
```

```
2 examples, 0 failures, 1 pending
```

# EXERCISE



## Faster Feedback While Developing Cookbooks

*The faster the feedback from our tests, the more likely we are to run them.  
The more likely we are to run them means they will catch more issues.*

### Objective:

- ✓ Review and run the existing tests
- ✓ Identify the tests that we need to write
- ❑ Write and execute the tests to identify the failure
- ❑ Fix the code and execute the tests to see success

# REFERENCE



## ChefSpec Documentation

Find within the documentation examples of testing for `include_recipe`.

- Search the README
- Search through the 'examples' directory

<https://github.com/chefspec/chefspec>



# Write the Test that Verifies the Include Recipe

 ~/apache/spec/unit/recipes/default\_spec.rb

```
# ... START OF THE SPEC FILE ...  
  it 'converges successfully' do  
    expect { chef_run }.to_not raise_error  
  end  
  
  it 'includes the install recipe' do  
    expect(chef_run).to include_recipe('apache::install')  
  end  
  
end  
end
```

# Execute the Tests to See the Failure



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

.F

failing example

Failures:

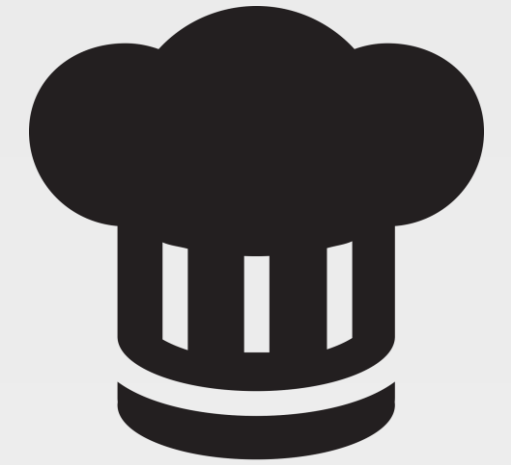
```
1) apache::default When all attributes are default, on an CentOS  
6.9 includes the install recipe
```

```
Failure/Error: expect(chef_run).to  
include_recipe('apache::install')
```

```
expected ["apache::default"] to include "apache::install"
```

```
# ./spec/unit/recipes/default_spec.rb:21:in `block (3 levels)  
in <top (required)>'
```

# EXERCISE



## Faster Feedback While Developing Cookbooks

*The faster the feedback from our tests, the more likely we are to run them.  
The more likely we are to run them means they will catch more issues.*

### Objective:

- ✓ Review and run the existing tests
- ✓ Identify the tests that we need to write
- ✓ Write and execute the tests to identify the failure
- ❑ Fix the code and execute the tests to see success

# Uncomment the Include Recipe



```
~/apache/recipes/default.rb
```

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



# Execute the Tests to See it Pass

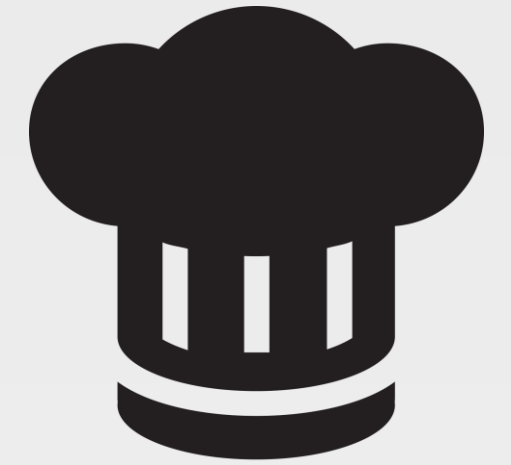


```
> chef exec rspec spec/unit/recipes/default_spec.rb
```

```
..
```

```
Finished in 0.67714 seconds (files took 4.26 seconds to load)  
2 examples, 0 failures
```

# EXERCISE



## Faster Feedback While Developing Cookbooks

*The faster the feedback from our tests, the more likely we are to run them.  
The more likely we are to run them means they will catch more issues.*

### Objective:

- ✓ Review and run the existing tests
- ✓ Identify the tests that we need to write
- ✓ Write and execute the tests to identify the failure
- ✓ Fix the code and execute the tests to see success



# Continue with Mutation Testing

- ☐ Comment out the next line in the apache cookbook's default recipe
- ☐ Write the example with expectation that will generate a failure
- ☐ Verify that one example generates a failure
- ☐ Restore the code in the recipe
- ☐ Verify that all examples pass

❖ Repeat this series of steps for each line within the default recipe

# Comment the Include Recipe



```
~/apache/recipes/default.rb
```

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



# Write the Test that Verifies the Include Recipe

```
~/apache/spec/unit/recipes/default_spec.rb
```

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

```
  it 'includes the install recipe' do
    expect(chef_run).to include_recipe('apache::install')
  end
```

```
  it 'includes the configuration recipe' do
    expect(chef_run).to include_recipe('apache::configuration')
  end
```

```
end
end
```

# Execute the Tests to See it Fail



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

```
...F
```

Failures:

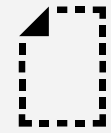
1) apache::default When all attributes are default, on an Centos 6.9 includes the configuration recipe

Failure/Error: expect(chef\_run).to  
include\_recipe('apache::configuration')

expected ["apache::default", "apache::install",  
"apache::service"] to include "apache::configuration"

# ./spec/unit/recipes/default\_spec.rb:27:in `block (3 levels)

# Uncomment the Include Recipe



```
~/apache/recipes/default.rb
```

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

# Execute the Tests to See it Pass



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

...

```
Finished in 0.97252 seconds (files took 4.33 seconds to load)  
3 examples, 0 failures
```

# Comment the Include Recipe



```
~/apache/recipes/default.rb
```

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

# Write the Test that Verifies the Include Recipe

```
~/apache/spec/unit/recipes/default_spec.rb
```

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

it 'includes the install recipe' do
  expect(chef_run).to include_recipe('apache::install')
end

it 'includes the configuration recipe' do
  expect(chef_run).to include_recipe('apache::configuration')
end

it 'includes the service recipe' do
  expect(chef_run).to include_recipe('apache::service')
end

end

end
```

# Execute the Tests to See it Fail



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

```
...F
```

Failures:

1) apache::default When all attributes are default, on an Centos 6.9 includes the service recipe

Failure/Error: expect(chef\_run).to  
include\_recipe('apache::service')

expected ["apache::default", "apache::install",  
"apache::configuration"] to include "apache::service"

# ./spec/unit/recipes/default\_spec.rb:31:in `block (3 levels)

# Uncomment the Include Recipe



```
~/apache/recipes/default.rb
```

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



# Execute the Tests to See it Pass



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

....

```
Finished in 1.38 seconds (files took 4.16 seconds to load)  
4 examples, 0 failures
```



# Continue with Mutation Testing

- ✓ Comment out the next line in the apache cookbook's default recipe
- ✓ Write the example with expectation that will generate a failure
- ✓ Verify that one example generates a failure
- ✓ Restore the code in the recipe
- ✓ Verify that all examples pass

❖ Repeat this series of steps for each line within the default recipe

# DISCUSSION



## Discussion

What functionality did you test in the integration tests?

What functionality did you test in these unit tests?

What do you see as the scope of unit testing versus integration testing?

What are the differences between a ChefSpec test and a InSpec test?

# 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**™

---