

WEBINAR

# WRITING ELEGANT TESTS

FRANKLIN WEBBER TRAINING AND TECHNICAL LEAD



# Franklin Webber

*Training and Technical Content Lead*

- ❖ [twitter.com/franklinwebber](https://twitter.com/franklinwebber)
- ❖ [github.com/burtlo](https://github.com/burtlo)
- ❖ [linkedin.com/in/fwebber](https://linkedin.com/in/fwebber)



Ceci n'est pas un entraîneur

# Who I Think You Are

## Chef Cookbook Author

- Author and maintain Chef Cookbooks with ChefSpec Tests
- Maintainer of a test suite that contains lots of duplication
- Interested in furthering your knowledge of Ruby's RSpec



# What You Will Learn

## The Focus

- ❖ Reduce duplication in our expectations, examples, and scenario setup
- ❖ Create re-useable test helpers
- ❖ Extract these testing resources into a Ruby gem



```
spec_helper.rb          default_spec.rb
1 require 'spec_helper'
2
3 describe 'ark::default' do
4   context 'when no attributes are specified, on an unspecified platform' do
5     let(:chef_run) do
6       runner = ChefSpec::SoloRunner.new
7       runner.converge(described_recipe)
8     end
9
10    it 'installs necessary packages' do
11      expect(chef_run).to install_package('libtool')
12      expect(chef_run).to install_package('autoconf')
13      expect(chef_run).to install_package('unzip')
14      expect(chef_run).to install_package('rsync')
15      expect(chef_run).to install_package('make')
16      expect(chef_run).to install_package('gcc')
17      expect(chef_run).to install_package('autogen')
18    end
19
20    it "does not install the gcc-c++ package" do
21      expect(chef_run).not_to install_package("gcc-c++")
22    end

```

# What You Will Learn

## The Focus

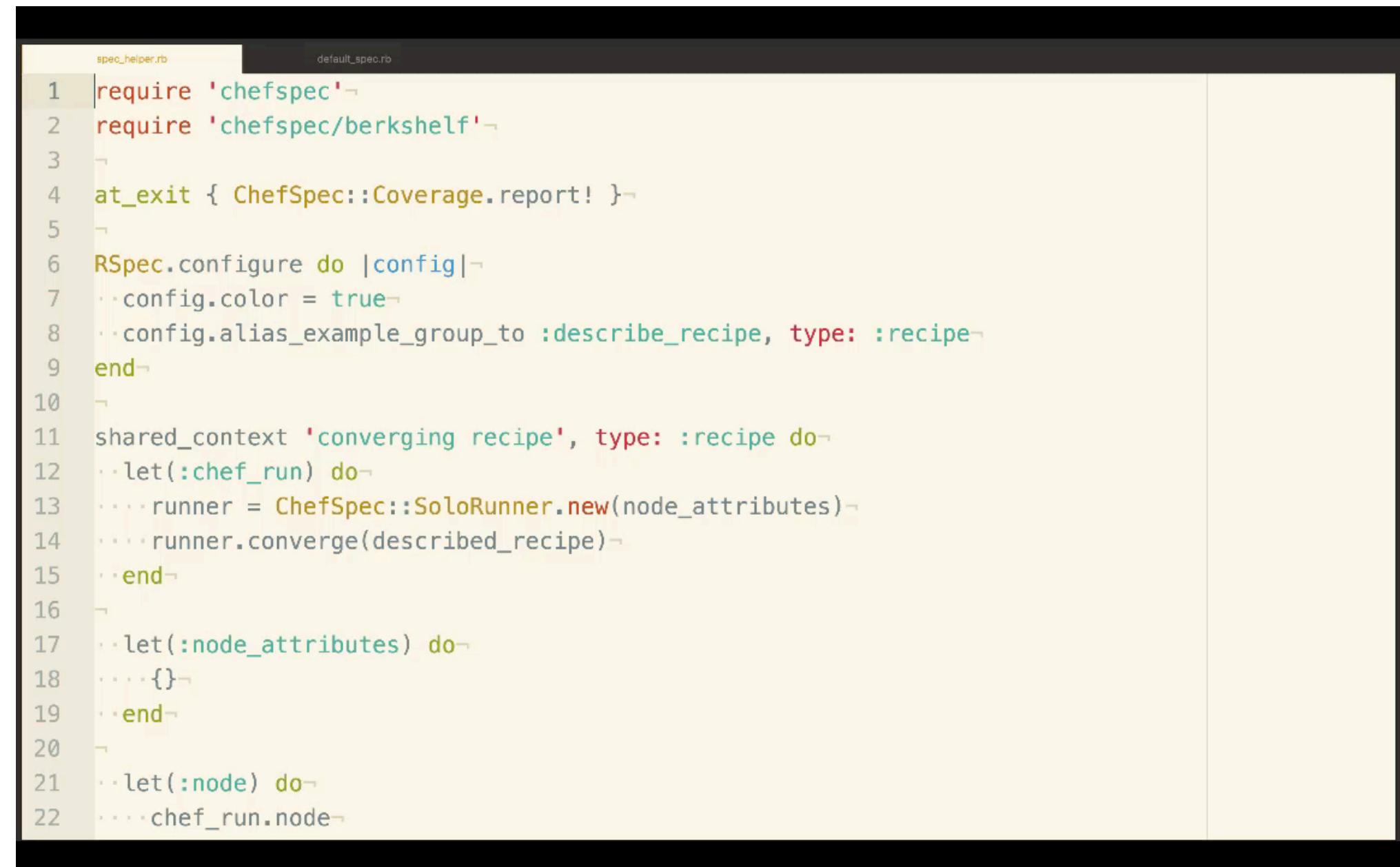
- ❖ Reduce duplication in our expectations, examples, and scenario setup
- ❖ **Create re-useable test helpers**
- ❖ Extract these testing resources into a Ruby gem

```
spec_helper.rb          default_spec.rb
1 require 'spec_helper'-
2 -
3 describe_recipe 'ark::default' do
4   context 'when no attributes are specified, on an unspecified platform' do
5     let(:installed_packages) do
6       %w[ libtool autoconf unzip rsync make gcc autogen ]
7     end
8
9     it_behaves_like 'installs packages'
10
11    let(:packages_not_installed) do
12      %w[ gcc-c++ seven_zip ]
13    end
14
15    it_behaves_like 'does not install packages'
16
17    it "apache mirror" do
18      expect(attribute('apache_mirror')).to eq "http://apache.mirrors.tds.net"
19    end
20
21    it "prefix root" do
22      expect(attribute('prefix_root')).to eq "/usr/local"
```

# What You Will Learn

## The Focus

- ❖ Reduce duplication in our expectations, examples, and scenario setup
- ❖ Create re-useable test helpers
- ❖ **Extract these testing resources into a Ruby gem**



A screenshot of a code editor showing RSpec ChefSpec configuration code. The code is in a file named `default_spec.rb`. It includes require statements for `'chefspec'` and `'chefspec/berkshelf'`, sets up coverage reporting, configures RSpec to use color output and alias examples for recipes, and defines shared contexts for converging recipes, node\_attributes, and nodes, along with nested let blocks for each.

```
1 |require 'chefspec'-
2 |require 'chefspec/berkshelf'-
3 |
4 |at_exit { ChefSpec::Coverage.report! }-
5 |
6 |RSpec.configure do |config|-
7 |  config.color = true-
8 |  config.alias_example_group_to :describe_recipe, type: :recipe-
9 |end-
10|
11|shared_context 'converging recipe', type: :recipe do-
12|  let(:chef_run) do-
13|    runner = ChefSpec::SoloRunner.new(node_attributes)-.
14|    runner.converge(described_recipe)-.
15|  end-
16|
17|  let(:node_attributes) do-
18|    {}-
19|  end-
20|
21|  let(:node) do-
22|    chef_run.node-
```

# Schedule

---

- ❖ **Introduction (5 minutes)**
- ❖ **techniques (40 minutes)**

*Introduce 7 concepts with demonstration, and review*

- ❖ **questions (12 minutes)**
- ❖ **wrap up (3 minutes)**

# Example Code Repository



```
> git clone https://github.com/chef-training/elegant_tests-repo.git
```

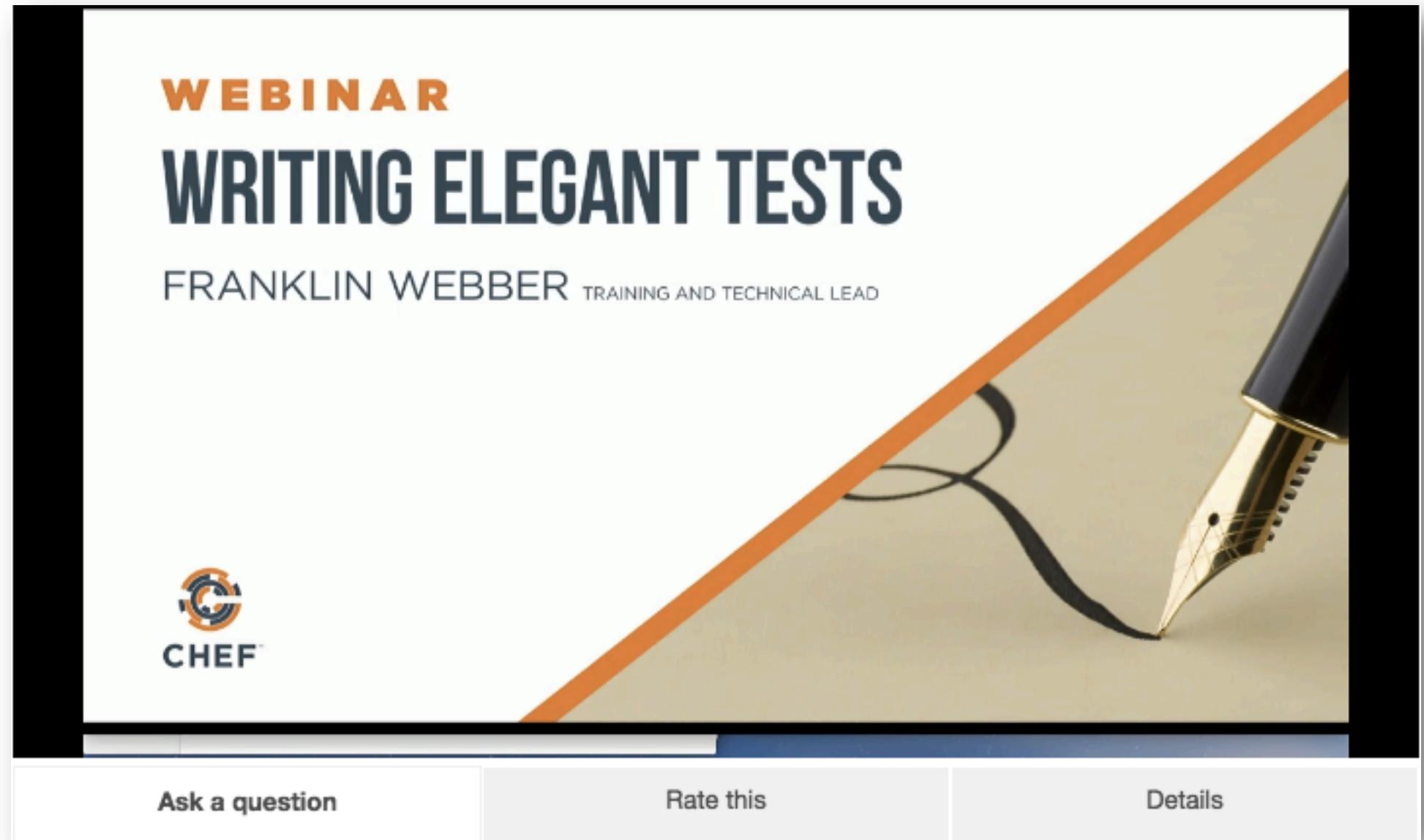
```
Cloning into 'elegant_tests-repo'...
remote: Counting objects: 67, done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 67 (delta 12), reused 67 (delta 12), pack-reused 0
Unpacking objects: 100% (67/67), done.
```

# Let Us Know What You Think

"Ask a Question" and I will answer it.

"Rate this" presentation to leave your feedback and help me do my work better.

To share, click on the appropriate professional/social network in "Details"



WEBINAR

# WRITING ELEGANT TESTS

FRANKLIN WEBBER TRAINING AND TECHNICAL LEAD



# Agenda

---

TECHNIQUES

- ❑ `let`
- ❑ `shared_examples`
- ❑ `def method`
- ❑ `shared_context`
- ❑ `require`
- ❑ `alias_example_group_to`
- ❑ `creating a Ruby gem`
- ❑ `questions`
- ❑ `wrap up`

# let

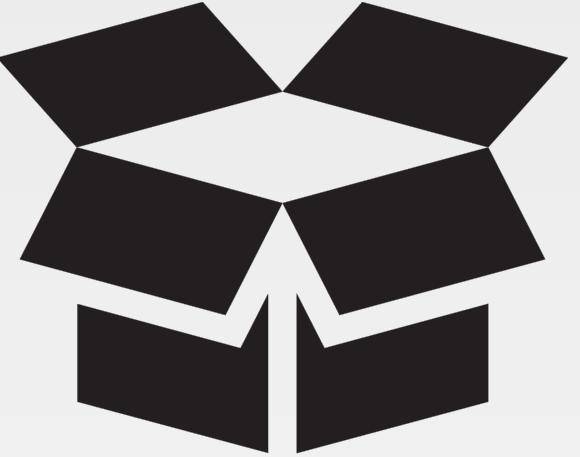
---

*"Since we cannot change reality, let us change the eyes which see reality."*

*~ Nikos Kazantzakis*

- Concepts
- Demonstration
- Review

# CONCEPT



## let

Use `let` to define a memoized helper method.

The value will be cached across multiple calls in the same example but not across examples. It is also lazy-evaluated: it is not evaluated until the first time the method it defines is invoked. See `let!` if you want to force the invocation before each example.

<https://goo.gl/BJp0IQ>

# Diagramming the let Helper Method

```
describe 'ark::default' do
  context 'when no attributes are ...' do
    1-let(:chef_run) do
      runner = ChefSpec::SoloRunner.new
      runner.converge(described_recipe)
      runner
    end

    it 'installs necessary packages' do
      4expect(chef_run).to install_p...
      expect(chef_run).to install_p...
    end

    it "does not install the gcc-c+..."
      expect(chef_run).not_to insta...
    end
  end
end
```

3

- 1 let is a RSpec helper method
- 2 Ruby Symbol
- 3 Code Block
- 4 Invocation

# Invoking the Helper Method

```
describe 'ark::default' do
  context 'when no attributes are ...'
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new
      runner.converge(described_recipe)
      runner
    end

    it 'installs necessary packages' do
      expect(chef_run).to install_p...
      expect(chef_run).to install_p...
    end
  end

  it "does not install the gcc-c+..."
    expect(chef_run).not_to insta...
  end

```

2

1

4

- 1 chef\_run sends a message
- 2 RSpec invokes the contents of the block
- 3 RSpec stores the contents of the execution
- 4 chef\_run sends a message
- 5 RSpec retrieves the stored execution

# Cached Within each Example

```
describe 'ark::default' do
  context 'when no attributes are ...'
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new
      runner.converge(described_recipe)
      runner
    end

    it 'installs necessary packages' do
      expect(chef_run).to install_p...
      expect(chef_run).to install_p...
    end

    it "does not install the gcc-c+..."
      expect(chef_run).not_to insta...
    end
  
```

- 1 chef\_run is loaded and stored
- 2 chef\_run uses the stored invocation
- 3 chef\_run is loaded and stored

# A chef\_run with Node Attributes

```
describe 'ark::default' do
  context 'when no attributes are specified, on CentOS' do
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new({ platform: 'centos',
                                         version: '6.7' })
      runner.converge(described_recipe)
    end
  end
end
```

# ... EXAMPLES WITHIN CONTEXT

# let

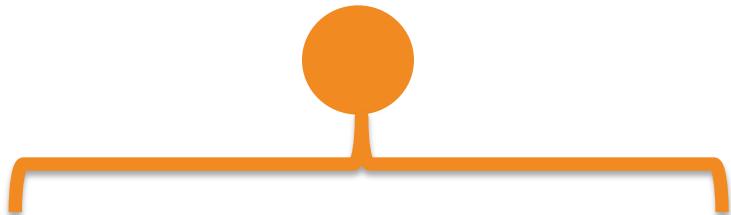
---

*"If the path be beautiful, let us not ask where it leads."*

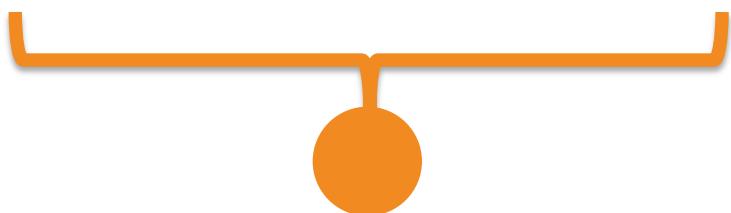
*~ Anatole France*

- Concepts
- Demonstration
- Review

# Objective



*Use `let` to express  
the tests succinctly.*



# let

---

"Let us always meet each other with smile, for the smile is the beginning of love."

~ Mother Teresa

- Concepts
- Demonstration
- Review

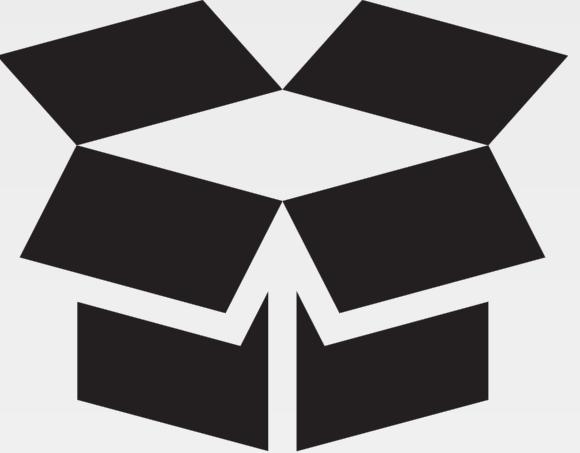
# EXAMPLE



## Example

<https://goo.gl/ChkP47>

# CONCEPT



## Using let for clarity

The use of the let to create the `chef_run` saves us from having to write the same code over-and-over again within each example.

We can define our own let helpers to increase the readability of our test code. Extracting important details and giving them a name.

<https://goo.gl/BJp0IQ>

# A `chef_run` with Node Attributes

```
describe 'ark::default' do
  context 'when no attributes are specified, on CentOS' do
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new({ platform: 'centos',
                                         version: '6.7' })
      runner.converge(described_recipe)
    end
  end
end
```

# ... EXAMPLES WITHIN CONTEXT

# Using let to Create Clearer Examples

```
describe 'ark::default' do
  context 'when no attributes are specified, on CentOS' do
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new(node_attributes)
      runner.converge(described_recipe)
    end

    let(:node_attributes) do
      { platform: 'centos', version: '6.7' }
    end

    # ... EXAMPLES WITHIN CONTEXT
  end
end
```

# let

---

"Let us not pray to be sheltered from dangers but to be fearless when facing them."

~ Rabindranath Tagore

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

## TECHNIQUES

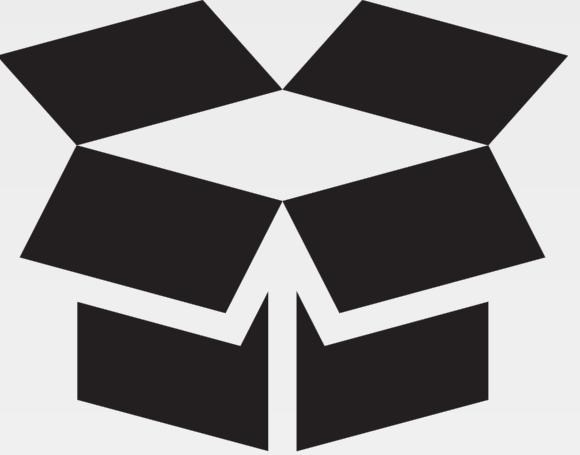
- ✓ `let`
- ❑ `shared_examples`
- ❑ `def method`
- ❑ `shared_context`
- ❑ `require`
- ❑ `alias_example_group_to`
- ❑ `creating a Ruby gem`
- ❑ `questions`
- ❑ `wrap up`

# shared\_examples

---

- Concepts
- Demonstration
- Review

# CONCEPT



## shared\_examples

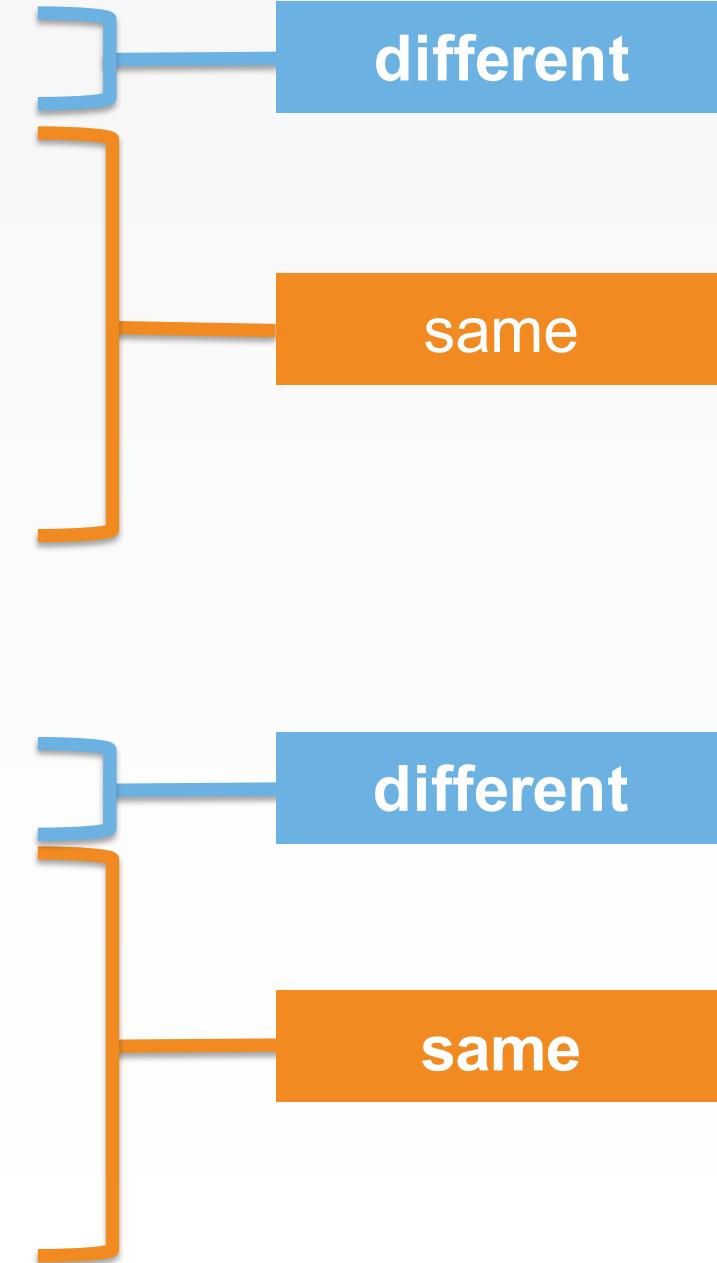
Shared examples let you describe behavior of classes or modules. When declared, a shared group's content is stored. It is only realized in the context of another example group, which provides any context the shared group needs to run.

<https://goo.gl/yi12tM>

# Finding Similar Expressed Expectations

```
context 'when no attributes are specified, on an unspecif...orm' do
  let(:installed_packages) ...
  it 'installs the necessary packages' do
    installed_packages.each do |name|
      expect(chef_run).to install_package(name)
    end
  end
end

context 'when no attributes are specified, on CentOS' do
  let(:installed_packages) ...
  it 'installs the necessary packages' do
    installed_packages.each do |name|
      expect(chef_run).to install_package(name)
    end
  end
end
```

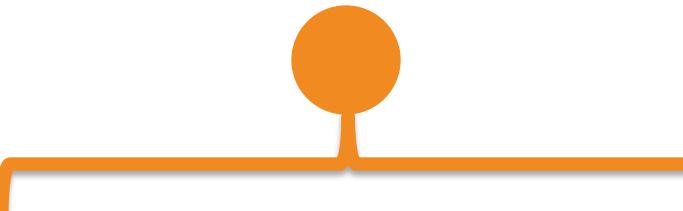


# shared\_examples

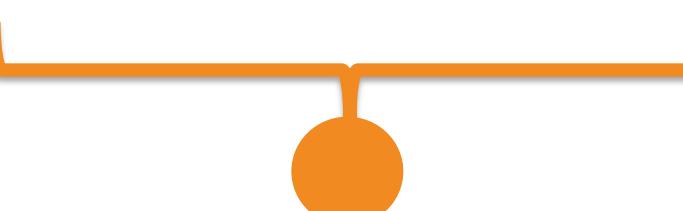
---

- Concepts
- Demonstration
- Review

# Objective



*Use shared examples to  
express similarities.*



# shared\_examples

---

- Concepts
- Demonstration
- Review

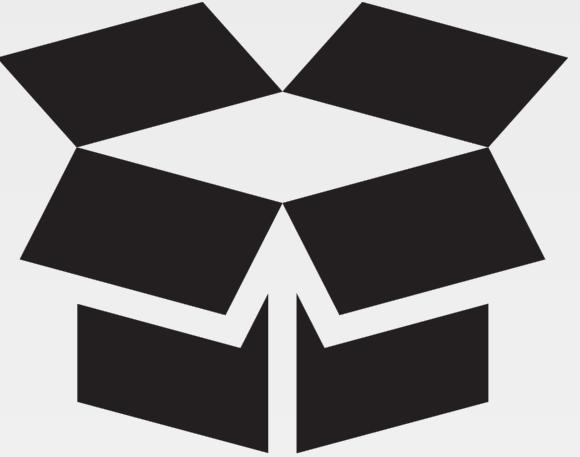
# EXAMPLE



## Example

<https://goo.gl/ChkP47>

# CONCEPT



## shared\_examples

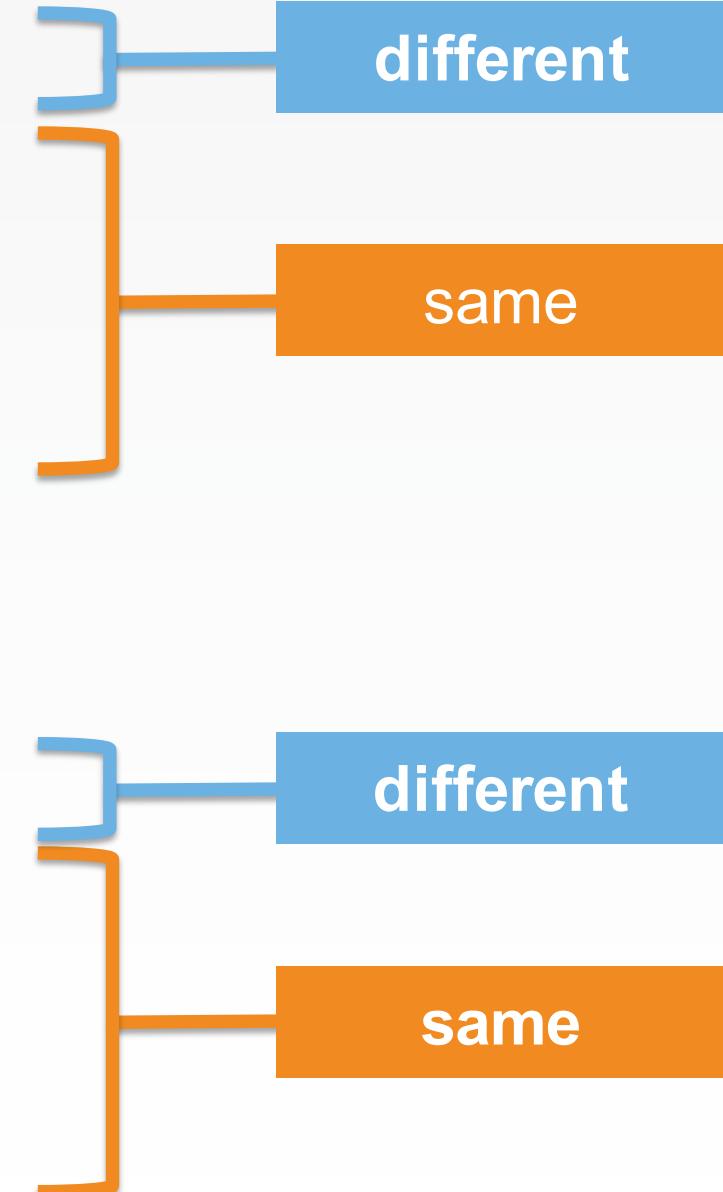
Shared examples let you describe behavior of classes or modules. When declared, a shared group's content is stored. It is only realized in the context of another example group, which provides any context the shared group needs to run.

<https://goo.gl/yi12tM>

# Finding Similar Expressed Expectations

```
context 'when no attributes are specified, on an unspecif...orm' do
  let(:installed_packages) ...
  it 'installs the necessary packages' do
    installed_packages.each do |name|
      expect(chef_run).to install_package(name)
    end
  end
end

context 'when no attributes are specified, on CentOS' do
  let(:installed_packages) ...
  it 'installs the necessary packages' do
    installed_packages.each do |name|
      expect(chef_run).to install_package(name)
    end
  end
end
```



# A Place to Share Examples

```
shared_examples 'installs packages' do
  it 'installs the necessary packages' do
    installed_packages.each do |name|
      expect(chef_run).to install_package(name)
    end
  end
end

context 'when no attributes are specified, on an unspecif...orm' do
  let(:installed_packages) ...
  it_behaves_like 'installs packages'
end

context 'when no attributes are specified, on CentOS' do
  let(:installed_packages) ...
  it_behaves_like 'installs packages'
end
```

# **shared\_examples**

---

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

## TECHNIQUES

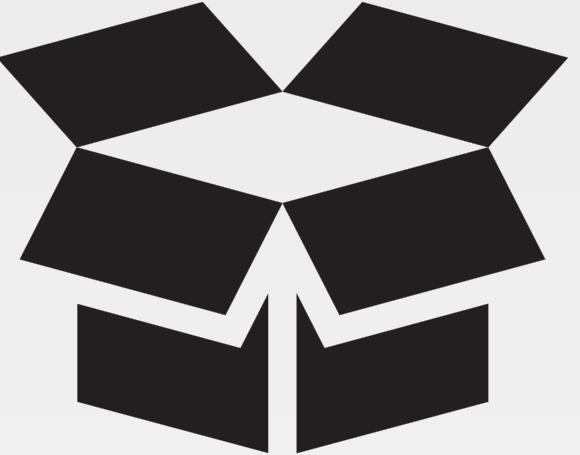
- ✓ **let**
- ✓ **shared\_examples**
- **def method**
- **shared\_context**
- **require**
- **alias\_example\_group\_to**
- **creating a Ruby gem**
- **questions**
- **wrap up**

# def method

---

- Concepts
- Demonstration
- Review

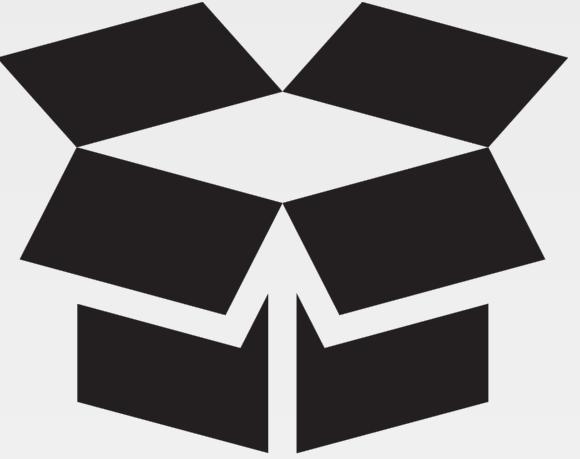
# CONCEPT



## Create helper method

You can define a Ruby method that takes care of some of the tedious work of retrieving node attributes.

# CONCEPT



## A Ruby Method with One Parameter

```
def file(name)
  # contents of method
  # last line automatically returns the value
end
```

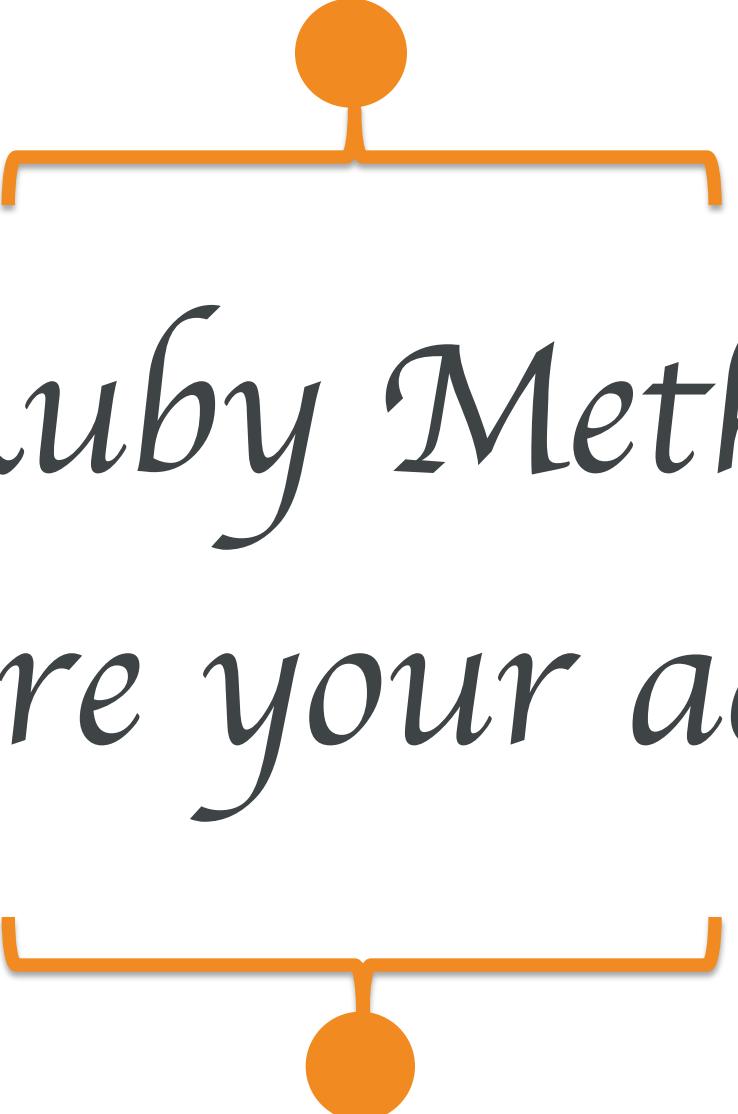
The method named 'file' has a single parameter named 'name'.

# def method

---

- Concepts
- Demonstration
- Review

# Objective



*Use Ruby Methods to  
capture your actions.*

# def method

---

- Concepts
- Demonstration
- Review

# EXAMPLE



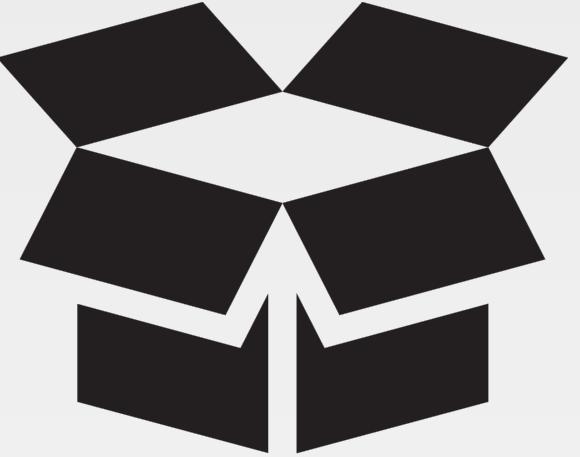
## Example

<https://goo.gl/9mRN1D>



(lowercase L)

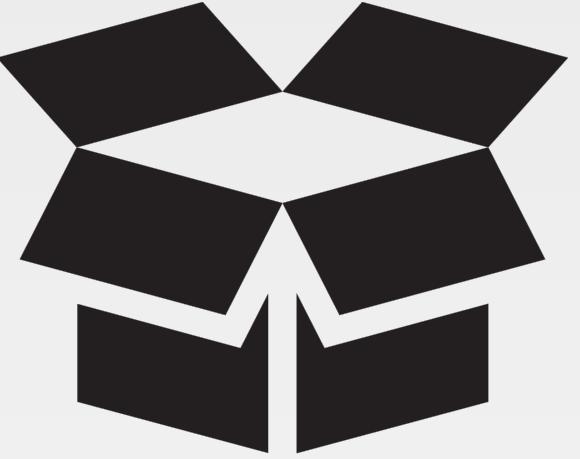
# CONCEPT



## Create helper method

You can define a Ruby method that takes care of some of the tedious work of retrieving node attributes.

# CONCEPT



## A Ruby Method with One Parameter

```
def file(name)
  # contents of method
  # last line automatically returns the value
end
```

The method named 'file' has a single parameter named 'name'.

# Viewing the Repetition of Retrieving Attributes

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

```
it "apache mirror" do
  attribute = chef_run.node['ark']['apache_mirror']
  expect(attribute).to eq "http://apache.mirrors.tds.net"
end

it "prefix root" do
  attribute = chef_run.node['ark']['prefix_root']
  expect(attribute).to eq "/usr/local"
end
```

# Refactoring with let to help ease the pain

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

```
let(:node) do
  chef_run.node
end

it "apache mirror" do
  attribute = node['ark']['apache_mirror']
  expect(attribute).to eq "http://apache.mirrors.tds.net"
end

it "prefix root" do
  attribute = node['ark']['prefix_root']
  expect(attribute).to eq "/usr/local"
end
```

# Implementing a Helper Method

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

```
let(:node) do
  chef_run.node
end

def attribute(name)
  node[described_cookbook][name]
end

it "apache mirror" do
  expect(attribute('apache_mirror')).to eq "http://apache.mirr....net"
end

it "prefix root" do
  expect(attribute('prefix_root')).to eq "/usr/local"
end
```

# def method

---

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

## TECHNIQUES

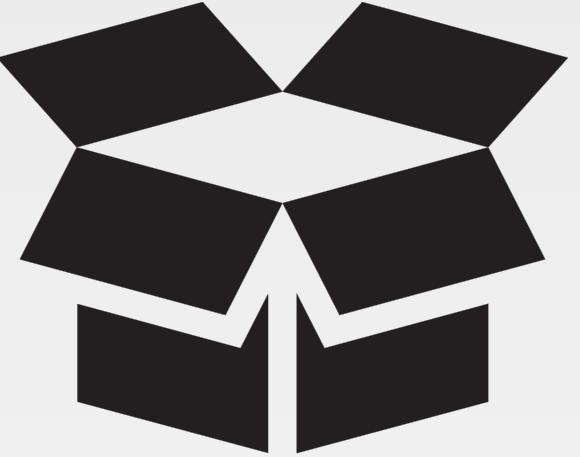
- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- **shared\_context**
- **require**
- **alias\_example\_group\_to**
- **creating a Ruby gem**
- **questions**
- **wrap up**

# shared\_context

---

- Concepts
- Demonstration
- Review

# CONCEPT



## shared\_context

Use `shared_context` to define a block that will be evaluated in the context of example groups either explicitly, using `include_context`, or implicitly by matching metadata.

<http://goo.gl/R0ujTA>

# shared\_context

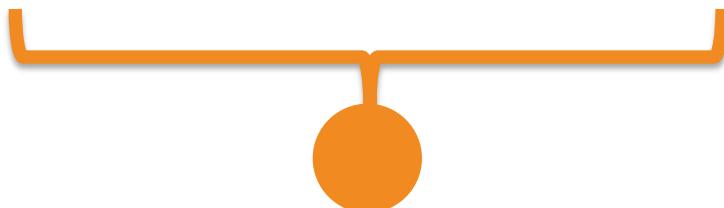
---

- Concepts
- Demonstration
- Review

# Objective



*Use **shared\_context**  
to save all that you wrote.*



# shared\_context

---

- Concepts
- Demonstration
- Review

# EXAMPLE



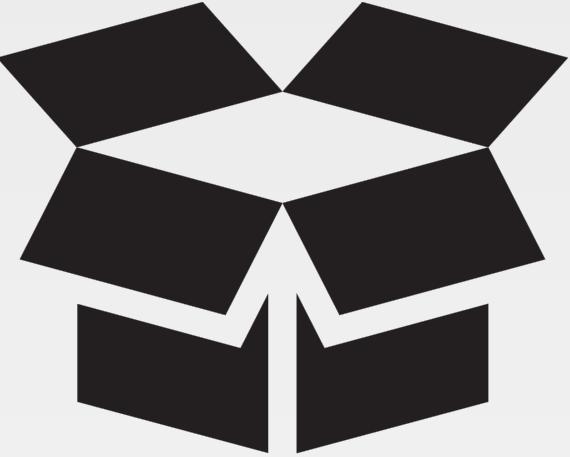
## Example

<https://goo.gl/9mRN1D>



(lowercase L)

# CONCEPT



## shared\_context

Use `shared_context` to define a block that will be evaluated in the context of example groups either explicitly, using `include_context`, or implicitly by matching metadata.

<http://goo.gl/R0ujTA>

# Repeating More than Examples

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

```
describe 'ark::default' do
  context 'when no attributes are specified, on an uns...platform' do
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new(node_attributes)
      runner.converge(described_recipe)
    end

    let(:node_attributes) do
      {}
    end

    def attribute(name) ...
  end
end
```

# Repeating More than Examples

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

```
shared_context 'converged recipe' do
  let(:chef_run) do
    runner = ChefSpec::SoloRunner.new(node_attributes)
    runner.converge(described_recipe)
  end

  let(:node_attributes) do
    {}
  end

  def attribute(name) ...
end

describe 'ark::default' do
  context 'when no attributes are specified, on an uns...platform' do
    include_context 'converged recipe'
```

# shared\_context

---

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

## TECHNIQUES

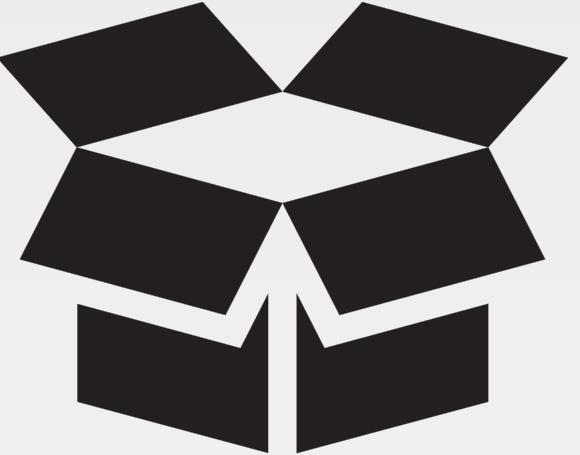
- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- **require**
- **alias\_example\_group\_to**
- **creating a Ruby gem**
- **questions**
- **wrap up**

# require

---

- Concepts
- Demonstration
- Review

# CONCEPT



## require

Loads the given filename, returning true if successful and false if the feature is already loaded.

If the filename does not resolve to an absolute path, it will be searched for in the directories listed in `$LOAD_PATH ($:)`.

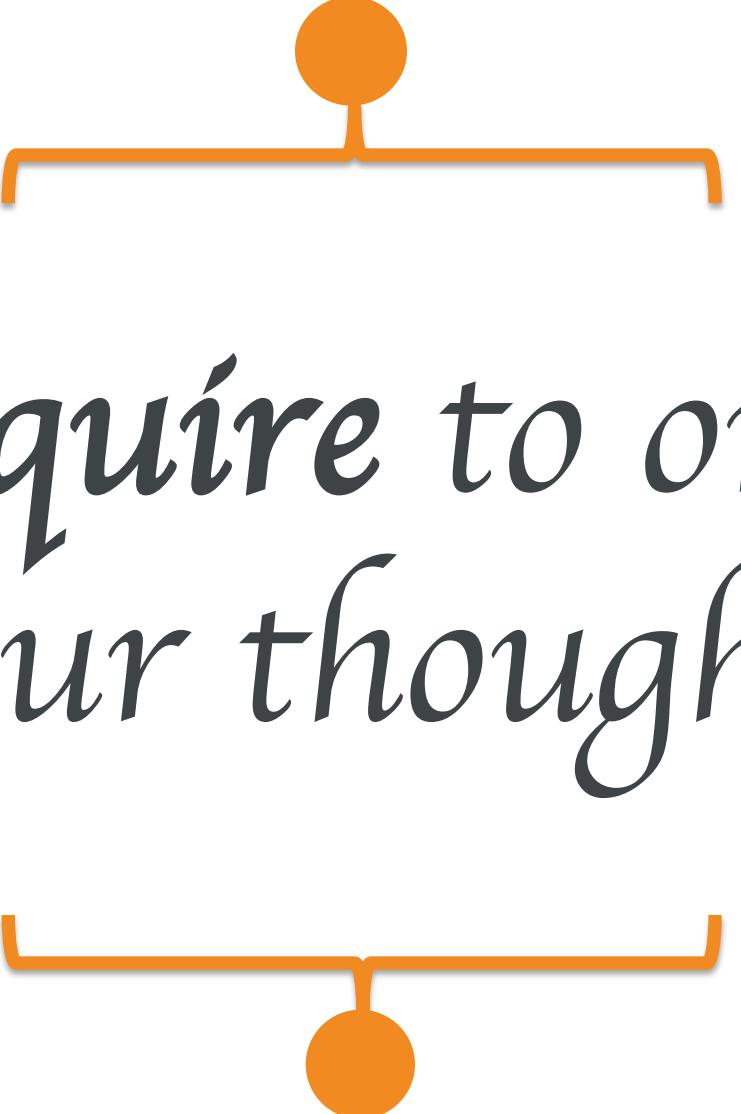
<http://goo.gl/cLKY37>

# require

---

- Concepts
- Demonstration
- Review

# Objective



*Use require to organize  
your thoughts.*

# require

---

- Concepts
- Demonstration
- Review

# EXAMPLE



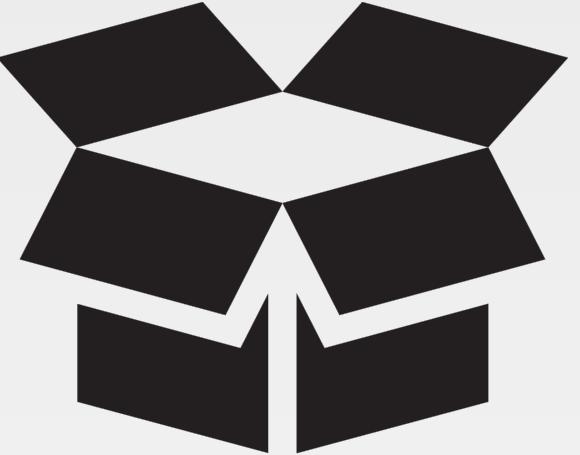
## Example

<https://goo.gl/9mRN1D>



(lowercase L)

# CONCEPT



## require

Loads the given filename, returning true if successful and false if the feature is already loaded.

If the filename does not resolve to an absolute path, it will be searched for in the directories listed in `$LOAD_PATH ($:)`.

<http://goo.gl/cLKY37>

# Requiring the spec\_helper file

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

```
require 'spec_helper'

describe 'ark::default' do
  context 'when no attributes are specified, on an ...platform' do
    let(:chef_run) do
      runner = ChefSpec::SoloRunner.new
      runner.converge(described_recipe)
    end
  end
end
```

# Viewing the spec\_helper file

~/ark/spec/spec\_helper.rb

```
require 'chefspec'
require 'chefspec/berkshelf'

at_exit { ChefSpec::Coverage.report! }

RSpec.configure do |config|
  config.color = true
end

# puts $LOAD_PATH
# ... define test helpers and content in this file
```

# Viewing the \$LOAD\_PATH



```
> chef exec rspec
```

```
/opt/chefdk/embedded/lib/ruby/gems/2.1.0/gems/uuidtools-2.1.5/lib
/Users/franklinwebber/ark/spec
/Users/franklinwebber/.chefdk/gem/ruby/2.1.0/gems/rspec-support-3.4.1/lib
/Users/franklinwebber/.chefdk/gem/ruby/2.1.0/gems/rspec-core-3.4.4/lib
/opt/chefdk/embedded/lib/ruby/gems/2.1.0/gems/net-ssh-3.1.1/lib
/opt/chefdk/embedded/lib/ruby/gems/2.1.0/gems/fauxhai-3.5.0/lib
/opt/chefdk/embedded/lib/ruby/gems/2.1.0/gems/diff-lcs-1.2.5/lib
/Users/franklinwebber/.chefdk/gem/ruby/2.1.0/gems/rspec-expectations-3.4.0/lib
/Users/franklinwebber/.chefdk/gem/ruby/2.1.0/gems/rspec-mocks-3.4.1/lib
/Users/franklinwebber/.chefdk/gem/ruby/2.1.0/gems/rspec-3.4.0/lib
```

# require

---

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

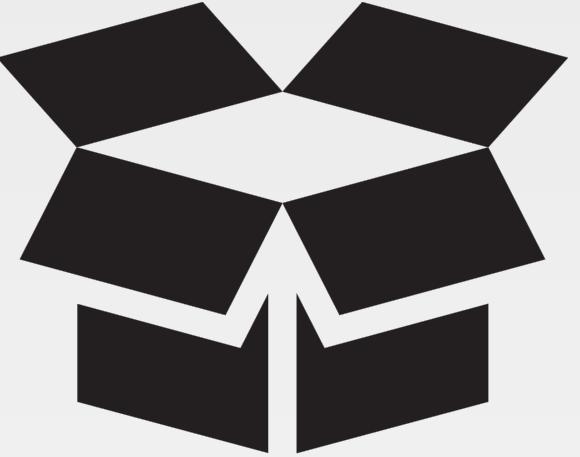
## TECHNIQUES

- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- ✓ **require**
- ☐ **alias\_example\_group\_to**
- ☐ **creating a Ruby gem**
- ☐ **questions**
- ☐ **wrap up**

# alias\_example\_group\_to

- Concepts
- Demonstration
- Review

# CONCEPT



## **alias\_example\_group\_to**

**describe** and **context** are the default aliases for **example\_group**. You can define your own aliases for **example\_group** and give those custom aliases default metadata.

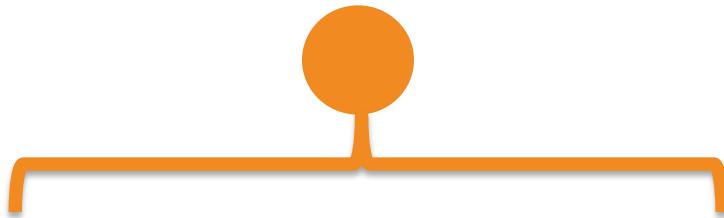
<http://goo.gl/DfUChL>

# alias\_example\_group\_to

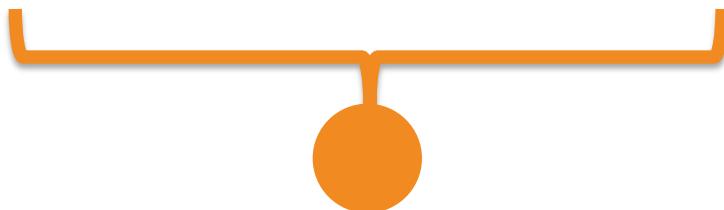
---

- Concepts
- Demonstration
- Review

# Objective



*Elegance comes when the context is assumed.*



# EXAMPLE



## Live Demonstration

<https://goo.gl/9mRN1D>



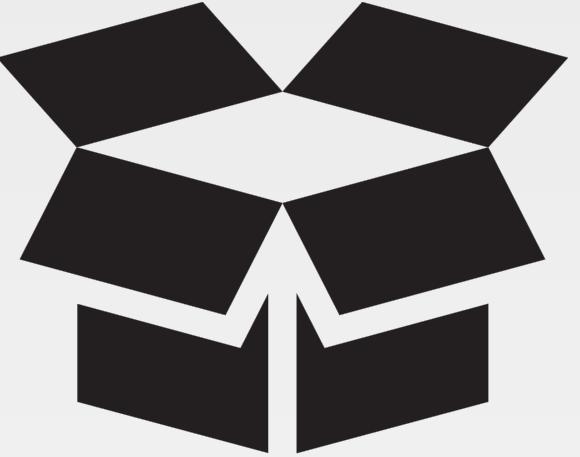
(lowercase L)

# alias\_example\_group\_to

---

- Concepts
- Demonstration
- Review

# CONCEPT



## **alias\_example\_group\_to**

**describe** and **context** are the default aliases for **example\_group**. You can define your own aliases for **example\_group** and give those custom aliases default metadata.

<http://goo.gl/DfUChL>

# Viewing a Spec with include\_context

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

```
describe 'ark::default' do
  context 'when no attributes are specified, on an ...platform' do
    include_context 'converged recipe'

    # ...
  end

  # ...
end
```

# Refactoring the Spec to auto include the context

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

```
describe_recipe 'ark::default' do
  context 'when no attributes are specified, on an uns...platform' do
    include_context 'converged recipe'

    # ... examples ...

  end

  # ... other contexts
end
```

# Defining the alias and tying it to the shared\_context

~/ark/spec/spec\_helper.rb

```
require 'chefspec'
require 'chefspec/berkshelf'

at_exit { ChefSpec::Coverage.report! }

RSpec.configure do |config|
  config.color = true
  config.alias_example_group_to :describe_recipe, :type => :recipe
end

shared_context 'converged recipe', :type => :recipe do
```

# alias\_example\_group\_to

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

## TECHNIQUES

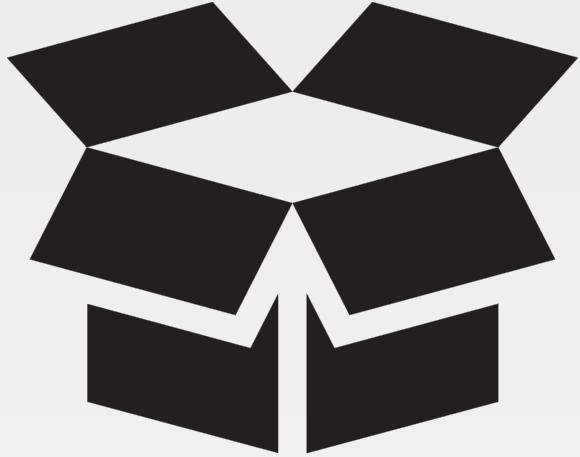
- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- ✓ **require**
- ✓ **alias\_example\_group\_to**
- creating a Ruby gem**
- questions**
- wrap up**

# creating a Ruby gem

---

- Concepts
- Demonstration

# CONCEPT

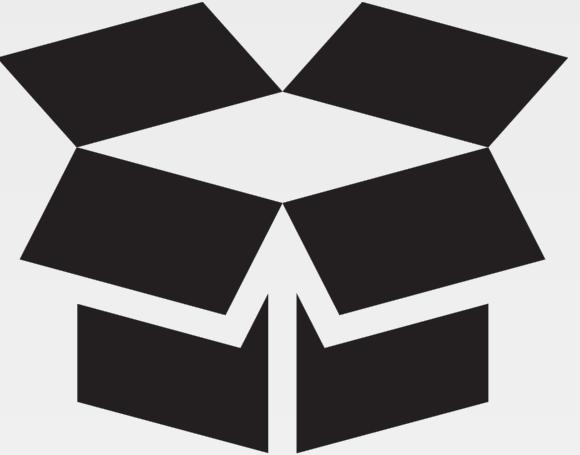


## RubyGems

RubyGems is a package manager for the Ruby programming language that provides a standard format for distributing Ruby programs and libraries (in a self-contained format called a "gem"), a tool designed to easily manage the installation of gems, and a server for distributing them.

<http://guides.rubygems.org/rubygems-basics>

# CONCEPT



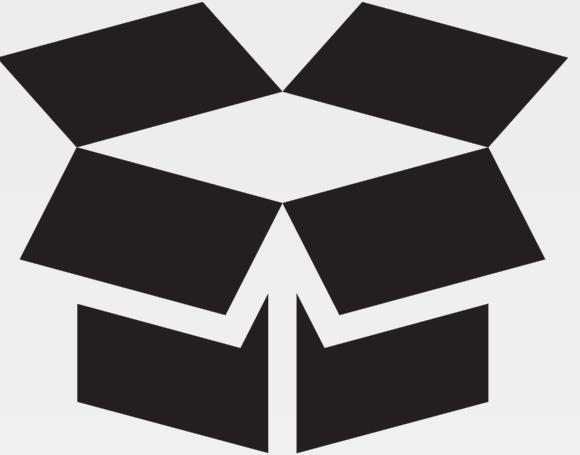
## gem

```
> chef gem --help
```

The command allows you build gems for distribution, install gems locally, and push gems to Gem server.

<http://guides.rubygems.org/command-reference>

# CONCEPT



## Bundler

Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed.

Bundler is an exit from dependency hell, and ensures that the gems you need are present in development, staging, and production. Starting work on a project is as simple as bundle install.

<http://bundler.io>

# CONCEPT



## bundle

```
> chef exec bundle help
```

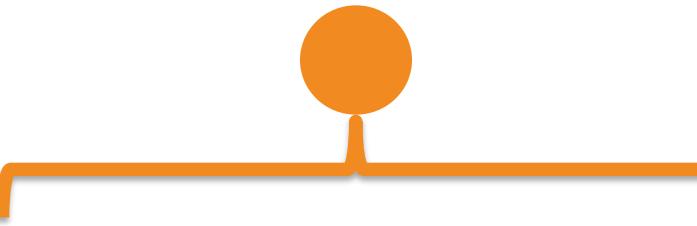
The command allows you to install and update a project's dependencies. It will also allow you to generate a cookbook.

# creating a Ruby gem

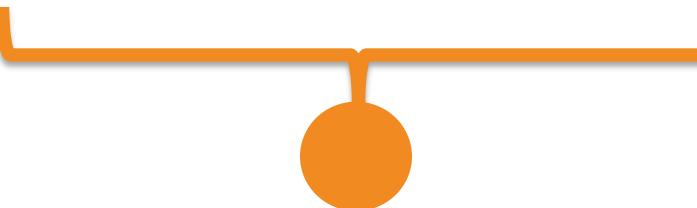
---

- Concepts
- Demonstration
- Review

# Objective



*Create Treasure!*



# creating a Ruby gem

---

- Concepts
- Demonstration
- Review

# EXAMPLE



## Example

[github.com/burtlo/chefspec-ohai](https://github.com/burtlo/chefspec-ohai)

# Generating the Gem



```
> chef exec bundle gem chefspec-myhelpers
```

```
Creating gem 'chefspect-myhelpers'...
  create  chefspec-myhelpers/Gemfile
  create  chefspec-myhelpers/.gitignore
  create  chefspec-myhelpers/lib/chefspect/myhelpers.rb
  create  chefspec-myhelpers/lib/chefspect/myhelpers/version.rb
  create  chefspec-myhelpers/chefspect-myhelpers.gemspec
  create  chefspec-myhelpers/Rakefile
  create  chefspec-myhelpers/README.md
  create  chefspec-myhelpers/bin/console
  create  chefspec-myhelpers/bin/setup
```

# Updating the Gem Specification

```
~/chefspec-myhelpers/chefspec-myhelpers.gemspec
```

```
Gem::Specification.new do |spec|  
  spec.name          = "chefspec-myhelpers"  
  spec.version       = Chefspec::Myhelpers::VERSION  
  spec.authors        = ["Franklin Webber"]  
  spec.email         = ["franklin.webber@gmail.com"]  
  
  spec.summary        = %q{Provides common ChefSpec Helpers.}  
  spec.description    = %q{ChefSpec Helpers that define an alias...}  
  spec.homepage       = "https://github.com/burtlo/chefspec...com"  
  spec.license        = "MIT"
```

# Moving the Shared Context to the Gem

~/chefspec-myhelpers/lib/chefspec/myhelpers.rb

```
RSpec.configure do |config|
  config.alias_example_group_to :describe_recipe, :type => :recipe
end

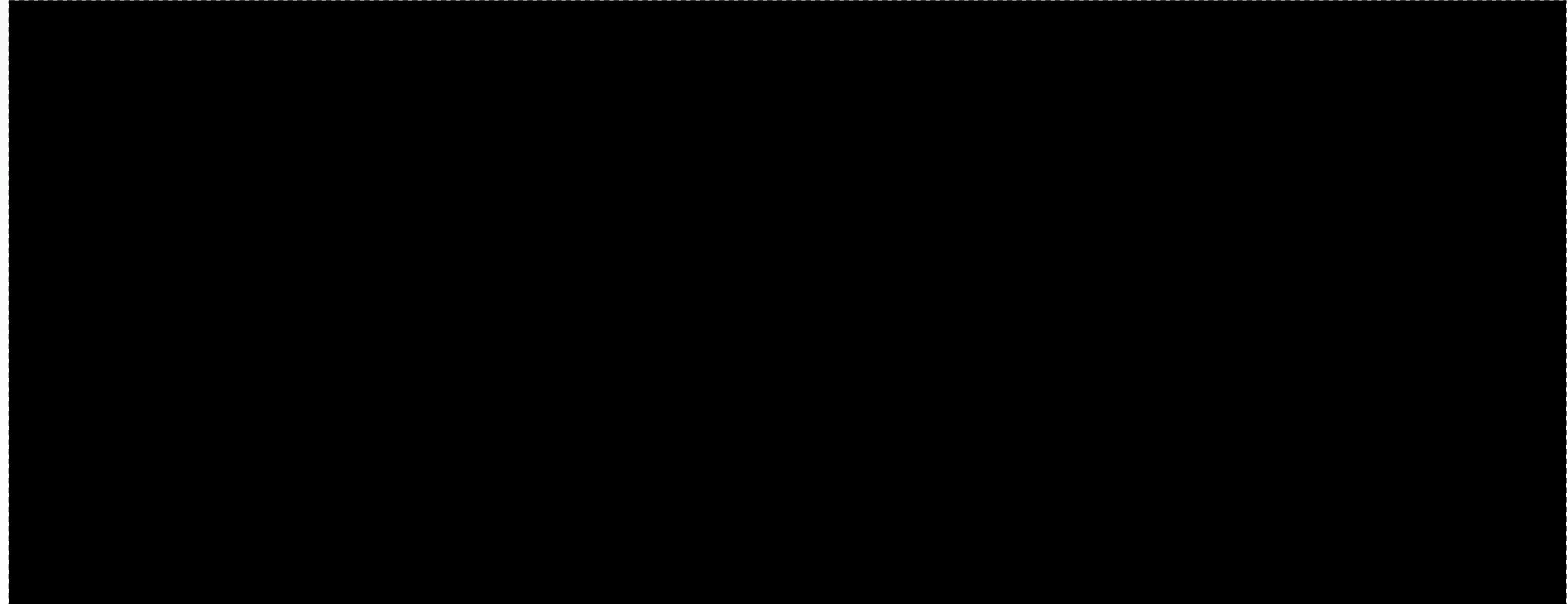
shared_context 'converged recipe' do
  let(:chef_run) do
    runner = ChefSpec::SoloRunner.new(node_attributes)
    runner.converge(described_recipe)
  end

  let(:node_attributes) do
    {}
  end
end
```

# Changing into the Gem's Directory



```
> cd chefspec-myhelpers
```



# Creating the Gem



```
> chef gem build chefspec-myhelpers.gemspec
```

Successfully built RubyGem

Name: `chefspec-myhelpers`

Version: `0.1.0`

File: `chefspec-myhelpers-0.1.0.gem`

# Installing the Gem



```
> chef gem install chefspec-myhelpers-0.1.0.gem
```

```
install chefspec-myhelpers-0.1.0.gem
Successfully installed chefspec-myhelpers-0.1.0
1 gem installed
```

# Listing the installed Gem



```
> chef gem list chefspec
```

```
*** LOCAL GEMS ***
```

```
chefspect (5.3.0)
```

```
chefspect-ohai (0.1.0)
```

```
chefspect-myhelpers (0.1.0)
```

# Updating the Cookbook's spec\_helper

~/ark/spec/spec\_helper.rb

```
require 'chefspec'  
require 'chefspec/berkshelf'  
require 'chefspec/myhelpers'
```

+

```
at_exit { ChefSpec::Coverage.report! }
```

```
RSpec.configure do |config|  
  config.color = true  
  config.alias_example_group_to :describe_recipe, :type => :recipe  
end
```

```
shared_context 'converged recipe', :type => :recipe do
```

-

# creating a Ruby gem

---

- ✓ Concepts
- ✓ Demonstration
- ✓ Review

# Agenda

---

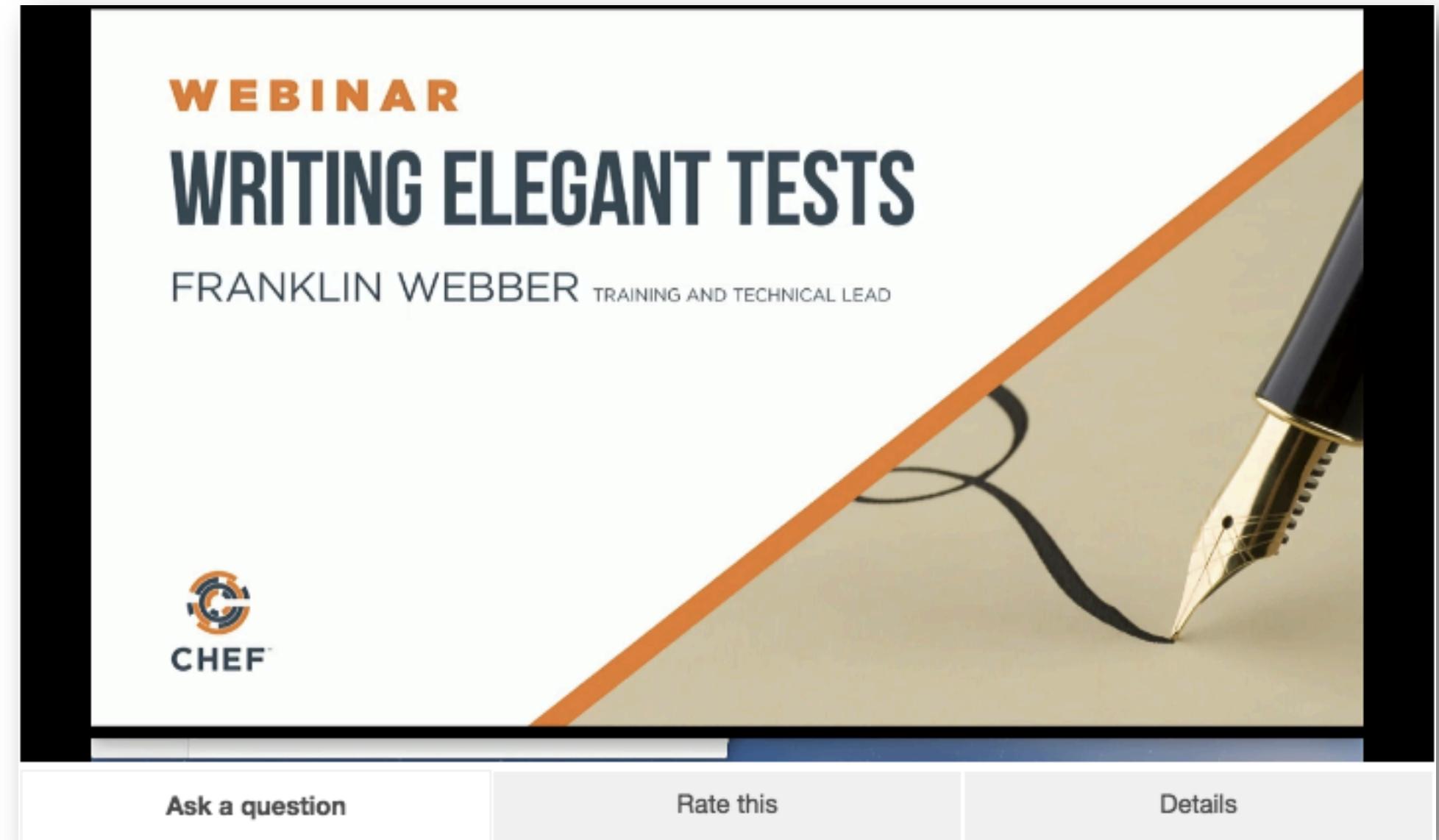
TECHNIQUES

- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- ✓ **require**
- ✓ **alias\_example\_group\_to**
- ✓ **creating a Ruby gem**
- questions**
- wrap up**

# Let Us Know What You Think

Email me at [franklin@chef.io](mailto:franklin@chef.io)

"Rate This" presentation to leave your feedback and help me do my work better.



# DISCUSSION



## Q&A

What questions can I answer for you?

# Agenda

---

## TECHNIQUES

- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- ✓ **require**
- ✓ **alias\_example\_group\_to**
- ✓ **creating a Ruby gem**
- ✓ **questions**
- **wrap up**

# Check The Attachments

All of the resources I am going to provide  
you can be found under "**Attachments**".



# Presentation Resources

## Slides

[https://github.com/chef-training/webinars/raw/master/writing\\_elegant\\_tests.pdf](https://github.com/chef-training/webinars/raw/master/writing_elegant_tests.pdf)

## Example Cookbook

[https://github.com/chef-training/elegant\\_tests-repo](https://github.com/chef-training/elegant_tests-repo)

# RSpec Resources

**let**

<https://relishapp.com/rspec/rspec-core/v/3-4/docs/helper-methods/let-and-let>

**shared\_examples**

<https://relishapp.com/rspec/rspec-core/v/3-4/docs/example-groups/shared-examples>

**shared\_context**

<https://relishapp.com/rspec/rspec-core/v/2-6/docs/example-groups/shared-context>

**alias\_example\_group\_to**

[http://www.rubydoc.info/github/rspec/rspec-core/RSpec%2FCore%2FConfiguration%3Aalias\\_example\\_group\\_to](http://www.rubydoc.info/github/rspec/rspec-core/RSpec%2FCore%2FConfiguration%3Aalias_example_group_to)

# Ruby Resources

## Ruby's require

<http://www.rubydoc.info/stdlib/core/Kernel%3Arequire>

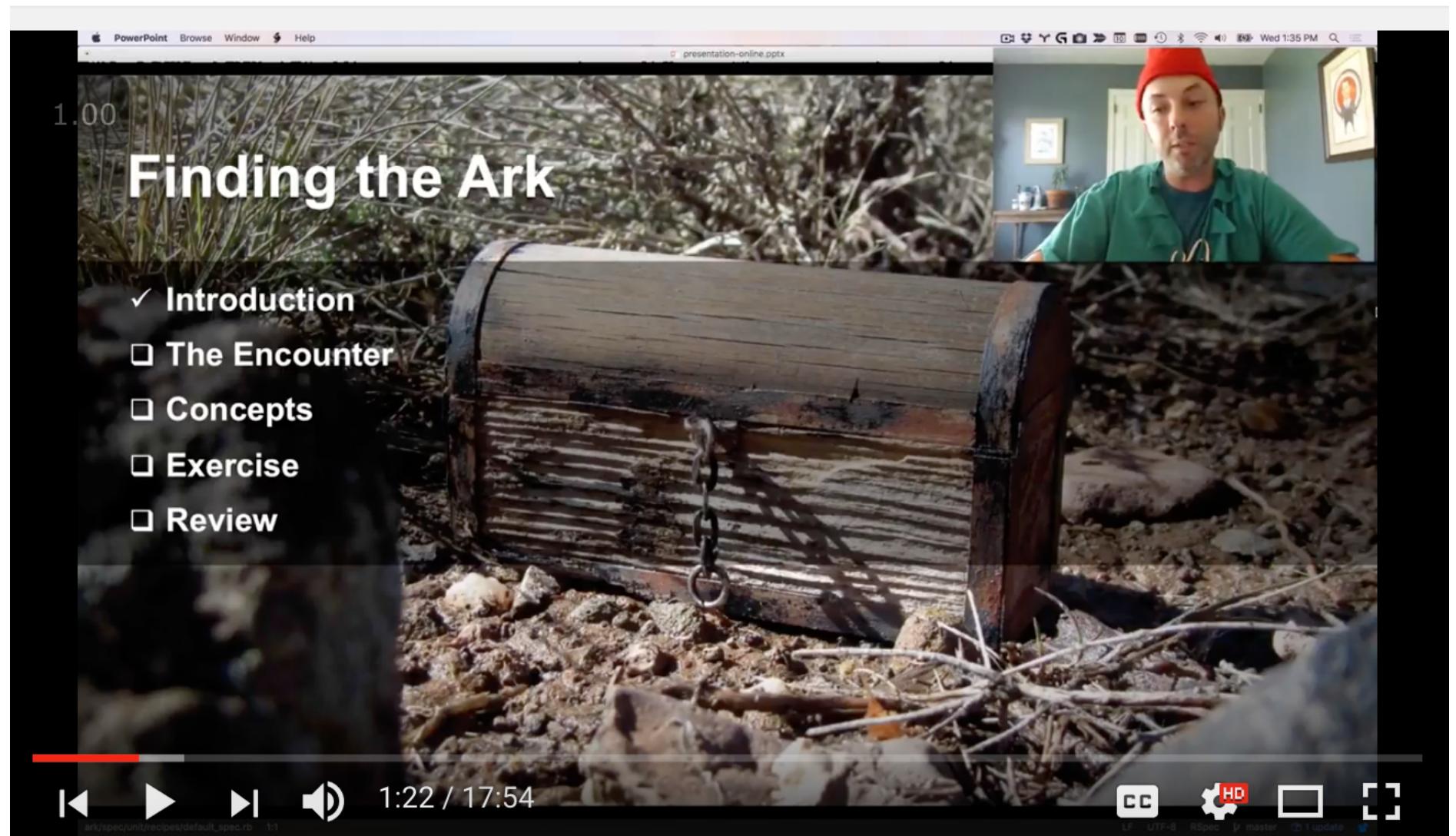
## Ruby Gems

<http://guides.rubygems.org/rubygems-basics/>

# Advanced Dungeons and Testing Dragons

This adventure will test your mettle as we delve through these dark corners of cookbook development.

<https://goo.gl/Qo2MfV>



# Agenda

---

TECHNIQUES

- ✓ **let**
- ✓ **shared\_examples**
- ✓ **def method**
- ✓ **shared\_context**
- ✓ **require**
- ✓ **alias\_example\_group\_to**
- ✓ **creating a Ruby gem**
- ✓ **questions**
- ✓ **wrap up**



CHEF™