

## Training Environment

### Chef Intermediate Topics

Your dedicated hands-on environment is just a click away.

We believe you shouldn't have to waste time copying gigabytes of software, shipping machines, or traveling, just to get your IT into people's hands.

When you click the 'Start Using' button to your right, you'll have instant access worldwide to a full, enterprise-grade IT environment, available through your browser, mobile device, or dedicated client and VPN connection. You can connect your existing datacenter servers, change configurations, load new software, and provide this functionality to others. It's just like an on-premises data center, but with more flexibility and none of the headaches of backup, access, replication, logging, SLAs, and other annoyances - we take care of all of that for you.

CloudShare's SaaS platform is a quick and easy way to share copies of your complex IT environments, online. So you can collaborate with customers, partners, and colleagues - for Demos, Proofs-of-Concept, Training, or other enterprise applications.

Questions? Click [here](#) to email this environment's author or click [here](#) for CloudShare support.



Chef Intermediate Topics

CentOS 6

**CentOS 6.3 Server**  
Description: OS: CentOS 6.3 x64  
Spec: 16 GB HD / 1 GB RAM  
OS: Linux  
State: Running  
[View details](#)

[View VM](#)

[Reboot VM](#) [Revert](#) [Delete](#)

Chef Intermediate Topics

CentOS 6

**CentOS 6.3 Server**  
Description: OS: CentOS 6.3 x64  
Spec: 16 GB HD / 1 GB RAM  
OS: Linux  
State: Running  
[Hide details](#)

[View VM](#)

[Reboot VM](#) [Revert](#) [Delete](#)

VM Details

External Address: [uv017z80u72k71dhkqt-vm.cld.sr](#)

Internal IP: 10.160.201.90

Total Memory: 1024 MB

Disk Size: 16 GB

CPU: 1

The machine was prepared in 2 minutes

Credentials ([show password](#))

Auto-login: root (local user)

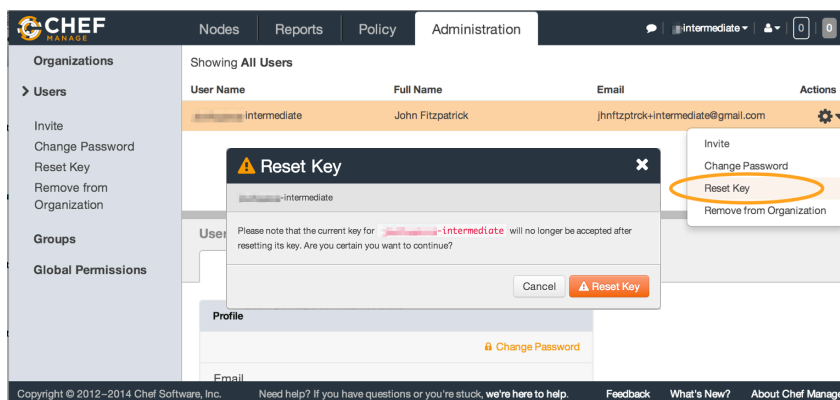
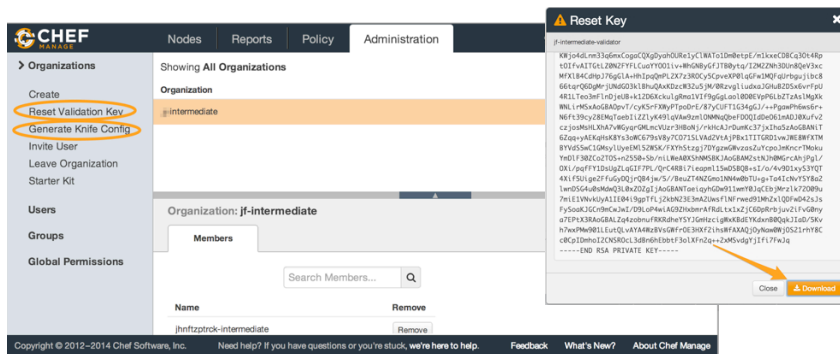
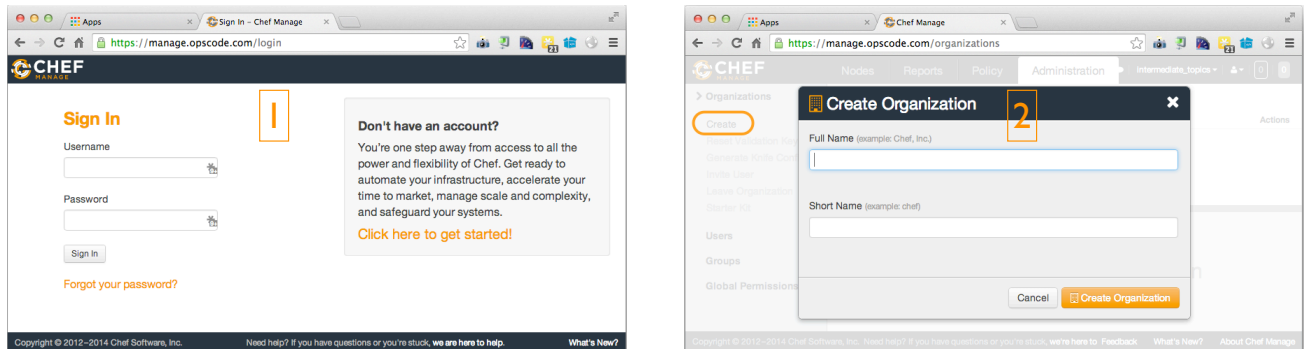
Username: root

Password: \*\*\*\*\*

```
$ ssh chef@<EXTERNAL_ADDRESS>
```

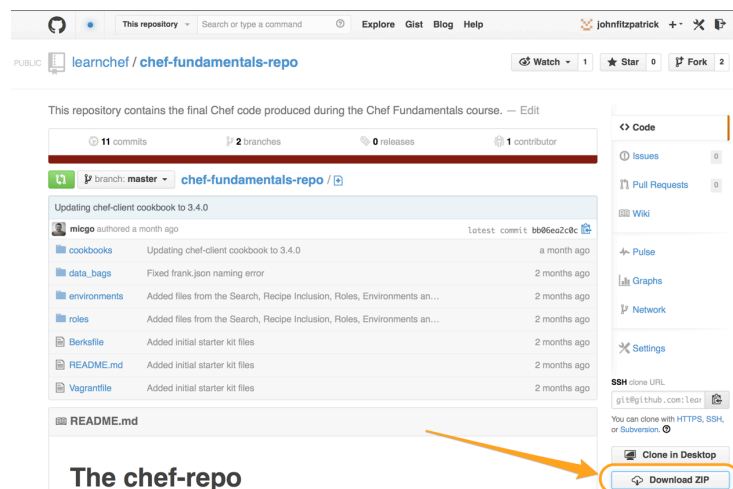
## Chef Fundamentals Refresher

### Create a new Org

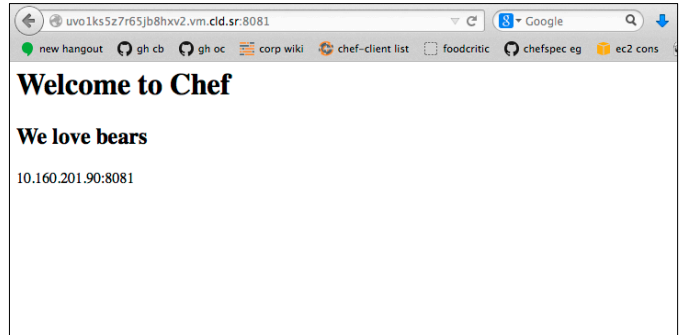
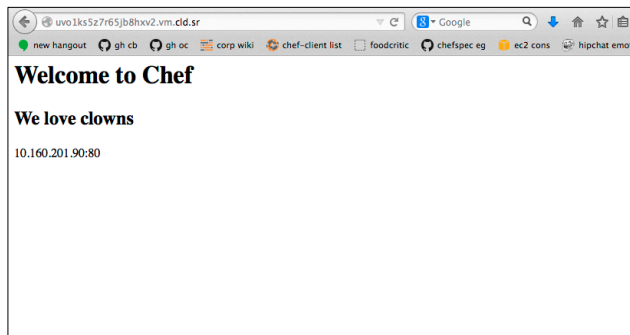


## Chef Intermediate

<https://github.com/learnchef/chef-fundamentals-repo>



```
$ cp ~/Downloads/chef-fundamentals-repo-master.zip .
$ unzip chef-fundamentals-repo-master.zip
$ cd chef-fundamentals-repo-master
$ ls -a
$ mkdir .chef
$ cp ~/Downloads/<yourname>.pem .chef
$ cp ~/Downloads/<your-org>-validator.pem .chef
$ cp ~/Downloads/knife.rb .chef
$ knife client list
$ knife cookbook upload -a
$ knife upload data_bags
$ knife role from file base.rb starter.rb webserver.rb
$ knife environment from file dev.rb production.rb
$ knife bootstrap <EXTERNAL_ADDRESS> --sudo -x chef -P chef -N
"node1" --bootstrap-version 11.12.8
$ ssh chef@<EXTERNAL_ADDRESS>
chef@node1:~$ ls /etc/chef
chef@node1:~$ which chef-client
chef@node1:~$ chef-client -v
$ knife node run_list set node1 'role[webserver]'
chef@node1$ sudo chef-client
chef@node1$ ps aux | grep chef-client
```



## Configuring Chef Clients

```
$ knife node show nodel
cookbooks/chef-client/recipes/config.rb
```

### **roles/base.rb**

```
name "base"
description "Base Server Role"
run_list "recipe[chef-client::delete_validation]",
"recipe[chef-client::config]", "recipe[chef-client]",
"recipe[ntp]", "recipe[motd]", "recipe[users]"
default_attributes(
  "chef_client" => {
    "config" => {
      "ssl_verify_mode" => ":verify_peer",
      "log_level" => ":info"
    }
  }
)
```

```
$ knife role from file base.rb
chef@nodel$ sudo chef-client
chef@nodel$ sudo cat /etc/chef/client.rb
$ knife ssl fetch
```

## Building Custom Resources

### **cookbooks/apache/metadata.rb**

```
maintainer      "apache"
maintainer_email "YOUR_EMAIL"
license         "All rights reserved"
description     "Installs/Configures apache"
long_description IO.read(File.join(File.dirname(__FILE__),
  'README.md'))
version       "0.3.0"
```

### **cookbooks/apache/recipes/default.rb**

### **cookbooks/apache/resources/vhost.rb**

```
actions :create, :remove
```

### **cookbooks/apache/providers/vhost.rb**

```
action :create do
  puts "My name is #{new_resource.name}"
end
```

### **cookbooks/apache/recipes/default.rb**

```
...
# Enable an Apache Virtualhost
apache_vhost "lions" do
  action :create
end
```

```
$ knife cookbook upload apache
chef@node1$ sudo chef-client
```

### **cookbooks/apache/providers/vhost.rb**

```
use_inline_resources
action :create do
  log "My name is #{new_resource.name}"
end
```

## Chef Intermediate

---

```
$ knife cookbook upload apache  
chef@node1$ sudo chef-client
```

### **cookbooks/apache/resources/vhost.rb**

```
actions :create, :remove  
attribute :site_name, :name_attribute => true, :kind_of =>  
String  
attribute :site_port, :default => 80, :kind_of => Fixnum
```

## Chef Intermediate

---

### **cookbooks/apache/providers/vhost.rb**

```
use_inline_resources
```

```
action :create do
  # Set the document root
  document_root = "/srv/apache/#{new_resource.site_name}"

  # Add a template for Apache virtual host configuration
  template "/etc/httpd/conf.d/#{new_resource.site_name}.conf" do
    source "custom.erb"
    mode "0644"
    variables(
      :document_root => document_root,
      :port => new_resource.site_port
    )
  end

  directory document_root do
    mode "0755"
    recursive true
  end

  # Add a template resource for the virtual host's index.html
  template "#{document_root}/index.html" do
    source "index.html.erb"
    mode "0644"
    variables(
      :site_name => new_resource.site_name,
      :port => new_resource.site_port
    )
  end
end
```

(Gist: <http://bit.ly/1o4X8Qi>)



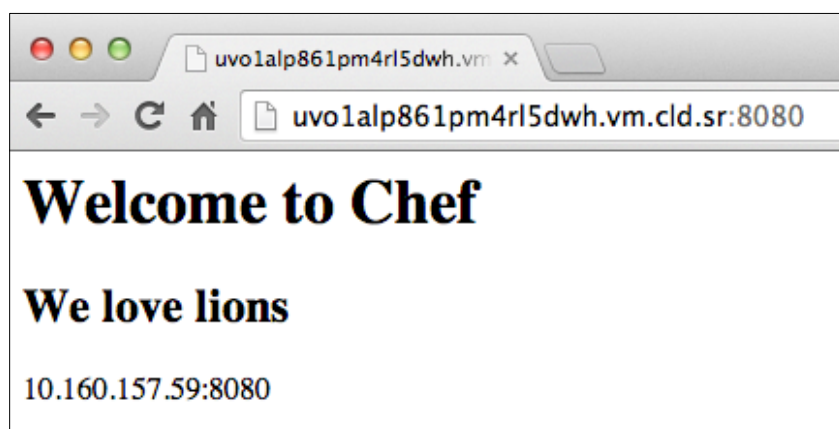
## Chef Intermediate

---

### **cookbooks/apache/recipes/default.rb**

```
# Enable an Apache Virtualhost
apache_vhost "lions" do
  site_port 8080
  action :create
  notifies :restart, "service[httpd]"
end
```

```
$ knife cookbook upload apache
chef@node1$ sudo chef-client
```



```
$ knife data_bag create apache_sites
$ mkdir -p data_bags/apache_sites
```

### **data\_bags/apache\_sites/clowns.json**

```
{
  "id": "clowns",
  "port": 80
}
```

```
$ knife data_bag from file apache_sites clowns.json
```

### **data\_bags/apache\_sites/bears.json**

```
{
  "id": "bears",
  "port": 81
}
```

```
$ knife data_bag from file apache_sites bears.json
```

```
data_bags/apache_sites/lions.json
```

```
{
  "id": "lions",
  "port": 8080
}
```

```
$ knife data_bag from file apache_sites lions.json
```

```
cookbooks/apache/recipes/default.rb
```

```
...
# Iterate over the apache sites
all_sites = search("apache_sites", ":*:")
all_sites.each do |site|
  # Enable an Apache Virtualhost
  apache_vhost site['id'] do
    site_port site['port']
    action :create
    notifies :restart, "service[httpd]"
  end
end
```

```
end
(Gist: http://bit.ly/1o4ZSNH )
```

```
$ knife cookbook upload apache
chef@node1$ sudo chef-client
```

```
cookbooks/apache/providers/vhost.rb
```

```
...
action :remove do
  file "/etc/httpd/conf.d/#{new_resource.site_name}.conf" do
    action :delete
  end
end
end
```

### **cookbooks/apache/recipes/default.rb**

```
# Disable the default virtual host
apache_vhost "welcome" do
  action :remove
  notifies :restart, "service[httpd]"
end
...
```

```
$ knife cookbook upload apache
chef@node1$ sudo chef-client
```

### chef-shell - The Chef Debugger

```
chef@node1$ sudo chef-shell -z
chef > help
chef > help(:nodes)
chef > node['fqdn']

chef > attributes_mode
chef:attributes > node.normal["interactive"] = "is cool"
chef:attributes > node["interactive"]

$ knife node show node1 -a interactive

chef:attributes > node.save

$ knife node show node1 -a interactive
$ knife node show node1 -Fj -l |more

chef:attributes > recipe_mode
chef:recipe > package "tree"
chef:recipe > run_chef
chef:recipe > resources
chef:recipe > resources("user[frank]")
chef:recipe > puts resources("user[frank]").to_text
chef:recipe > exit

chef > reset
chef > nodes.list
chef > nodes.all {|n| "#{n.name}:#{n.chef_environment}" }
chef > nodes.transform(:name => "node1") {|n|
n.chef_environment("dev")}
chef > nodes.all {|n| "#{n.name}:#{n.chef_environment}" }

$ knife exec -E 'nodes.transform(:name => "node1") {|n|
n.chef_environment("production")} '

$ knife search node "*:*" -a chef_environment
```

## Chef Intermediate

---

```
chef > nodes.all {|n| "#{n.name}:#{n.chef_environment}"}
```

```
$ knife node environment set node1 _default
```

```
$ knife raw
```

```
https://api.opscode.com/organizations/yourorg/nodes/node1 |  
grep chef_environment
```

## Writing Ohai Plugins

```
chef@node1:~$ ohai
chef@node1:~$ ohai -v
chef@node1:~$ apachectl -t -D DUMP_MODULES
```

```
$ knife cookbook site download ohai 2.0.0
$ tar -zxvf ohai-2.0.0.tar.gz -C cookbooks/
$ knife cookbook upload ohai
```

**cookbooks/ohai/attributes/default.rb**

**cookbooks/apache/metadata.rb**

```
name                  'apache'
maintainer             'Your Name'
maintainer_email      'your\_email@example.com'
license               'All rights reserved'
description            'Installs/Configures apache'
long_description      IO.read(File.join(File.dirname(__FILE__),
'README.md'))
version               '0.4.0'
depends 'ohai'
```

**apache/files/default/plugins/modules.rb**

```
Ohai.plugin(:Apache) do
  require 'mixlib/shellout'
  provides "apache/modules"

  collect_data(:default) do
    modules = Mixlib::ShellOut.new("apachectl -t -D DUMP_MODULES")
    modules.run_command
    apache Mash.new
    apache[:modules] = modules.stdout
  end
end
```

### **cookbooks/apache/recipes/ohai\_plugin.rb**

```
ohai 'reload_apache' do
  plugin 'apache'
  action :nothing
end

cookbook_file "#{node['ohai']['plugin_path']}/modules.rb" do
  source 'plugins/modules.rb'
  owner  'root'
  group  'root'
  mode   '0755'
  notifies :reload, 'ohai[reload_apache]', :immediately
end
include_recipe 'ohai::default'

$ knife cookbook upload apache
$ knife node run_list set node1
'recipe[apache::ohai_plugin'],'role[base'],'recipe[apache]'
chef@node1:~$ sudo chef-client
$ knife node show node1 -a apache
chef@node1:~$ cat /etc/chef/ohai_plugins/modules.rb
```

## Chef Intermediate

---

### **apache/files/default/plugins/modules.rb**

```
Ohai.plugin(:Apache) do
  provides "apache/modules"

  collect_data(:default) do
    apache Mash.new
    apache[:modules] = {:static => [], :shared => []}
    modules = shell_out("apachectl -t -D DUMP_MODULES")
    modules.stdout.each_line do |line|
      fullkey, value = line.split(/\(/, 2).map {|i| i.strip}
      apache_mod = fullkey.gsub(/_module/, "")
      dso_type = value.to_s.chomp("\)")
      if dso_type.match(/shared/)
        apache[:modules][:shared] << apache_mod
      elsif dso_type.match(/static/)
        apache[:modules][:static] << apache_mod
      end
    end
  end
end

(Gist: http://bit.ly/1jGAsmB)
```

```
$ knife cookbook upload apache
chef@node1:~$ sudo chef-client
```

```
$ knife node show node1 -a apache.modules
```

```
chef@node1$ /opt/chef/bin/chef-shell
```

```
chef > Ohai::Config['plugin_path'] << "/etc/chef/ohai_plugins"
```

```
chef > o = Ohai::System.new
chef > o.all_plugins
```

```
chef > o.refresh_plugins(attribute_filter='apache/modules')
```

```
$ knife node show node1 -a etc.passwd
```



### **roles/base.rb**

```
name "base"
description "Base Server Role"
run_list "recipe[chef-client::delete_validation]",
"recipe[chef-client::config]", "recipe[chef-client]",
"recipe[ntp]", "recipe[motd]", "recipe[users]"
default_attributes(
  "chef_client" => {
    "config" => {
      "ssl_verify_mode" => ":verify_peer"
      "log_level" => ":info"
    }
  },
  "ohai" => {
    "disabled_plugins" => [":Passwd"]
  }
)
```

```
$ knife role from file base.rb
```

```
chef@node1:~$ sudo chef-client && sudo chef-client
```

```
$ knife node show node1 -a etc.passwd
```

```
chef@node1:~$ sudo cat /etc/chef/client.rb
```

### Chef Client Run Internals

#### **apache/recipes/default.rb**

```
...
ruby_block "randomly_choose_language" do
  block do
    if Random.rand > 0.5
      node.run_state['scripting_language'] = 'php'
    else
      node.run_state['scripting_language'] = 'perl'
    end
  end
end

package "scripting_language" do
  package_name lazy { node.run_state['scripting_language'] }
  action :install
end

...

$ knife cookbook upload apache
chef@node1$ sudo chef-client
```

## Implementing Chef Handlers

```
$ knife cookbook site download chef_handler 1.1.6
$ tar -zxvf chef_handler-1.1.6.tar.gz -C cookbooks/
$ knife cookbook upload chef_handler
```

```
$ knife cookbook create email_handler
```

### **cookbooks/email\_handler/recipes/default.rb**

```
chef_gem "pony" do
  action :install
end
include_recipe "chef_handler"
```

### **cookbooks/email\_handler/recipes/default.rb**

```
cookbook_file
"#{node['chef_handler']['handler_path']}/email_handler.rb" do
  source "handlers/email_handler.rb"
  owner "root"
  group "root"
  mode "0644"
end

chef_handler "MyCompany::EmailMe" do
  source "#{node['chef_handler']['handler_path']}/email_handler.rb"
  arguments [node['email_handler']['from_address'],
             node['email_handler']['to_address']]
  action :enable
end
(Gist: http://bit.ly/T22uzb)
```

### **cookbooks/email\_handler/attributes/default.rb**

```
default['email_handler']['from_address'] = "chef@localhost"
default['email_handler']['to_address'] = "chef@localhost"
```

```
../email_handler/files/default/handlers/email_handler.rb
```

## Chef Intermediate

---

```
require 'rubygems'
require 'pony'

module MyCompany
  class EmailMe < Chef::Handler
    def initialize(from_address, to_address)
      @from_address = from_address
      @to_address = to_address
    end

    def report
      status = "Failed"
      if success?
        status = "Successful"
      end

      subject = "#{status} Chef run report from #{node.name}"
      report_string = ""

      # report on changed resources
      if ! run_status.updated_resources.empty?
        # get some info about all the changed resources!
        run_status.updated_resources.each do |r|
          report_string += "The resource #{r.name} was changed in
cookbook #{r.cookbook_name} at #{r.source_line}\n"
        end
      else
        report_string += "No resources changed by chef-client\n"
      end

      Pony.mail(:to => @to_address,
               :from => @from_address,
               :subject => subject,
               :body => report_string)
    end
  end
end
```

## Chef Intermediate

---

(Gist <http://bit.ly/T25AD2> )

```
$ knife cookbook site download postfix 3.1.8
$ tar -zxvf postfix-3.1.8.tar.gz -C cookbooks/
$ knife cookbook upload postfix
```

```
$ knife cookbook create mailx
```

```
cookbooks/mailx/recipes/default.rb
```

```
package "mailx" do
  action :install
end
```

```
cookbooks/email_handler/recipes/default.rb
```

```
chef_gem "pony" do
  action :install
end
```

```
include_recipe "chef_handler"
include_recipe "postfix"
include_recipe "mailx"
```

```
cookbooks/email_handler/metadata.rb
```

```
name          "email_handler"
maintainer    "You"
maintainer_email "you@somewhere.com"
license       "Apache 2.0"
description   "Email me on every Chef run"
long_description IO.read(File.join(File.dirname(__FILE__),
  'README.md'))
version       "0.1.0"
depends "chef_handler"
depends "postfix"
depends "mailx"
```

## Chef Intermediate

---

```
$ knife cookbook upload email_handler postfix mailx
```

### **roles/base.rb**

```
name "base"
description "Base Server Role"
run_list "recipe[email_handler]",
"recipe[chef-client::delete_validation]",
"recipe[chef-client::config]", "recipe[chef-client]",
"recipe[ntp]", "recipe[motd]", "recipe[users]"
```

```
$ knife role from file base.rb
chef@node1$ sudo chef-client
chef@node1$ mail
```

## Cookbook Style & Correctness

```
$ gem install foodcritic --no-ri --no-rdoc -v 3.0.3
$ foodcritic cookbooks/apache
$ foodcritic cookbooks/apache -t ~FC003 -t ~FC009

$ gem install rubocop --no-ri --no-rdoc -v 0.18.1
$ rubocop cookbooks
$ rubocop cookbooks/apache
$ rubocop cookbooks/apache --auto-gen-config
$ echo "inherit_from: rubocop-todo.yml" >> .rubocop.yml
```

### **cookbooks/apache/.rubocop.yml**

```
AlignParameters:
  Enabled: false
```

```
Encoding:
  Enabled: false
```

```
LineLength:
  Max: 200
```

```
SingleSpaceBeforeFirstArg:
  Enabled: false
```

```
$ rubocop cookbooks/apache
```

### **apache/files/default/plugins/modules.rb**

```
...
modules.stdout.each_line do |line|
  fullkey, value = line.split(/\(/, 2).map { |i| i.strip }
  apache_mod = fullkey.gsub(/_module/, "")
  ...
```

```
$ rubocop cookbooks/apache
```



### An Introduction to ChefSpec

```
$ gem install chefspec -v 4.0.1
```

```
cookbooks/motd/spec/spec_helper.rb
```

```
require 'chefspect'
```

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

```
$ mkdir cookbooks/motd/spec
```

```
$ mkdir cookbooks/motd/spec/unit
```

```
cookbooks/motd/spec/unit/default_spec.rb
```

```
require_relative '../spec_helper.rb'
```

```
describe 'motd::default' do
```

```
  let(:chef_run) { ChefSpec::Runner.new.converge(described_recipe) }
```

```
  it 'does something' do
```

```
    skip 'need to write this test'
```

```
  end
```

```
end
```

```
$ rspec cookbooks/motd
```

## Chef Intermediate

---

### **cookbooks/motd/spec/unit/default\_spec.rb**

```
require_relative '../spec_helper.rb'

describe 'motd::default' do
  let(:chef_run) { ChefSpec::Runner.new.converge(described_recipe) }

  it 'creates an motd correctly' do
    expect(chef_run).to create_template('/etc/motd').with(
      :user => 'root',
      :group => 'root',
      :mode => '0644'
    )
  end
end

$ rspec cookbooks/motd
```

### **cookbooks/motd/recipes/default.rb**

```
...
template "/etc/motd" do
  source "motd.erb"
  mode "0644"
  owner "root"
  group "root"
end

$ rspec cookbooks/motd
$ knife cookbook upload apache motd
```

### **cookbooks/mailx/spec/spec\_helper.rb**

```
require 'chefspec'
at_exit { ChefSpec::Coverage.report! }
```

## Chef Intermediate

---

### **cookbooks/mailx/spec/unit/default\_spec.rb**

```
require_relative '../spec_helper'

describe 'mailx::default' do
  context 'on Debian' do
    let(:chef_run) { ChefSpec::Runner.new({:platform =>
'ubuntu', :version => '14.04'}).converge(described_recipe) }

    it 'should install the correct packages' do
      expect(chef_run).to install_package 'mailutils'
    end
  end

  context 'on CentOS' do
    let(:chef_run) { ChefSpec::Runner.new({:platform =>
'centos', :version =>
'6.5'}).converge(described_recipe) }

    it 'should install the correct packages' do
      expect(chef_run).to install_package 'mailx'
    end
  end
end
```

(Gist <http://bit.ly/1tJiSRI>)

```
$ rspec cookbooks/mailx
```

### **cookbooks/mailx/attributes/default.rb**

```
case node['platform_family']
when "debian"
  default['mailutils']['mailx-package'] = "mailutils"
when "rhel"
  default['mailutils']['mailx-package'] = "mailx"
end
```

### **cookbooks/mailx/recipes/default.rb**

```
package node['mailutils']['mailx-package'] do
  action :install
end
```

```
$ rspec cookbooks/mailx
```