Following is a quick-start for getting a Chef and Test Kitchen environment set up to start writing and testing Chef cookbooks.

**Development Environment Set-up:**  
  
Unless you are a Ruby developer and already have a complete environment set-up the easiest thing to do is to install the Chef Development Kit, <https://downloads.chef.io/chef-dk/>.  
  
After downloading and installing the package add a CHEF\_HOME and RUBY\_HOME var pointed to the embedded version of Ruby that comes with the Chef SDK.  
  
  export CHEF\_HOME=/opt/chefdk  
  export RUBY\_HOME=/opt/chefdk/embedded  
  PATH="$CHEF\_HOME/bin:$RUBY\_HOME/bin:$PATH"  
  
**Building Cookbooks:**  
  
Generate the cookbook skeleton:  
  
$ chef generate cookbook <name-of-cookbook>  
  
Then cd into that directory and initialize the cookbook:  
  
$ kitchen init --create-gemfile  
  
You will also want to run bundle install, which will require your user to have sudo permissions.  
  
$ bundle install  
  
Run kitchen converge to verify before starting dev.

$ kitchen converge <name-of-vm>

Create a recipe

$ chef generate recipe <name-of-recipe>

[Optional] To use attributes, first create the attributes directory structure

$ chef generate attribute default  
  
Some helpful Tips:  
  
1.  Answer the following questions for each new cookbook you are going to create.  It will help you focus the purpose of the cookbook:  
  
. Name  
. Purpose  
. Success criteria  
. App/Service  
. Required steps  
  
2.  Creating template files:  
  
$ chef generate template <name-of-file>  
  
    . Sample variable to be added to a template:  
      <%= node['hostname'] %>  
  
3.  Available host variables.  You can see all of the variables available to be used in templates by logging in to a chef managed VM and issuing the following command:

$ kitchen login <name-of-vm>  
  
$ ohia | less  
  
4.  Add a IP to a kitchen VM so that additional kitchen VMs can talk to each other.  Add the following to your .kitchen.yml file.  
  
platforms:  
  - name: centos-7.1  
    driver:  
      network:  
      - ["private\_network", {type: "dhcp"}]

or

platforms:  
  - name: centos-7.1  
    driver:  
      network:  
      - ["private\_network", {ip: "192.168.33.10"}]

When specifying an IP, check the vboxnet0 interface on the host machine to see what network the VirtualBox host is configured and pick an IP in that range.

In my case, I had to make sure that only one machine was configured under 'platform', or that each had a network configuration.

Also, I had to have 'require\_chef\_omnibus: true' under the vagrant driver as follows:

driver:  
  name: vagrant  
  require\_chef\_omnibus: true

5.  Debugging Attributes:  Using the 'pp' module you can format the output of the node.debug('<attribute-name>').

Add the following to a recipe, and run it with kitchen converge:

require 'pp'  
  
node.default['ipaddress'] = '1.1.1.1'  
pp node.debug\_value('ipaddress')