Chef-solo & Recipe

Update the ubuntu

```
# apt-get update && apt-get -y upgrade
```

Install chef by using omnibus installer

```
# curl -L https://www.opscode.com/chef/install.sh | bash
```

check the version

```
# chef-solo -v
```

should setup a file structure that will help us organise our various Chef files. Opscode, the makers of Chef provide one. They call it simply the Chef Repository.

```
# wget http://github.com/opscode/chef-repo/tarball/master

# tar -zxf master

# mv chef-chef-repo* chef-repo


# rm master
```

go to the chef-repo

```
# cd chef-repo/
```

First we should tell knife where to find our cookbooks directory.

```
# mkdir .chef

# echo "cookbook_path [ '/root/chef-repo/cookbooks' ]" > .chef/knife.rb
```

create a cookbook for apache2 as an example

```
# knife cookbook create apache2
```

```
** Creating cookbook phpapp
** Creating README for cookbook: phpapp
** Creating CHANGELOG for cookbook: phpapp
** Creating metadata for cookbook: phpapp
```

# nano /root/chef-repo/cookbooks/apache2/recipes/default.rb

```
#
# Cookbook Name:: apache2
# Recipe:: default
#
# Copyright 2016, YOUR_COMPANY_NAME
#
# All rights reserved - Do Not Redistribute
#


package 'apache2' do
        action 'install'
end

service 'apache2' do
        action 'restart'
end

file '/var/www/html/index.html' do
        content 'hello'
        notifies :restart, 'service[apache2]'
end
```

run the following command for run recipe in client

# chef-client --local-mode --runlist 'recipe[apache2]'

link :::

http://gettingstartedwithchef.com/first-steps-with-chef.html

********************************Recipe************************************

to install package

```
package 'package-name'  do
        action 'install'
end
```

*****************************Recipe********************************

to start the service

```
service 'service-name' do
        action 'start'        # action [ :enable, :start ] for start at booting.
end
```

*****************************Recipe********************************

to create a file

```
file '/path/file-name'  do
        content 'some-data'
        owner 'root'
        group 'root'
        mode '777'
end
```

*****************************Recipe********************************

to add some file from master,

use
cookbook_file instead of file in above
and add some file in files/default directory of cookbook

or

template instead of file in above
and add some file in templates/default directory of cookbook

```
cookbook_file '/path/file-name'  do
        source 'file-name-in /files/default/ '
        owner 'root'
        group 'root'
        mode '777'
end


template '/path/file-name'  do
        source 'file-name-in /templates/default/ '
        owner 'root'
        group 'root'
        mode '777'
end
```

********************************Recipe************************************

to create directory

```
directory '/path/dir-name'  do
        owner 'root'
        group 'root'
        mode '777'
end
```

********************************Recipe************************************

to create a group

```
group 'group-name' do
        action 'create'
        gid '1500'
end
```

********************************Recipe************************************

to create a user

```
user 'user-name'  do
        action 'create'
        uid '1500'
        gid '1500'                # use group if create a group already with this gid
        password 'any'
        home '/home/user-name'
        shell '/bin/bash'
end
```

********************************Recipe************************************

to notify

```
notifies :restart, 'service[apache2]'           # if there is a service in the name of apache2
```

********************************Recipe************************************

to execute a command

```
execute 'update-upgrade' do
  command 'apt-get update && apt-get upgrade -y'
  environment 'path' => '/bin:/sbin:/usr/bin:/usr/sbin'          # optional, will work without path also
  action 'run'
end
```

***************************Recipe******************************

notifies	# notify some command when change in this command
subscribes	# notify this command when change in another command

:before
:immediate
:delayed

eg:

notifies :action, 'resource[name]', :timer	# action specifies for that command not for this command

notifies :restart, 'file[/var/www/html/index.html]', :immediate

subscribes :action, 'resource[name]', :timer	# action specifies for this command not for that command

subscribes :restart, 'file[/var/www/html/index.html]', :immediate

***************************Recipe******************************

notifies

Ruby Type: Symbol, ‘Chef::Resource[String]’

A resource may notify another resource to take action when its state changes. Specify a 'resource[name]', the :action that resource should take, and then the :timer for that action. A resource may notifiy more than one resource; use a notifies statement for each resource to be notified.

A timer specifies the point during the chef-client run at which a notification is run. The following timers are available:

:before
: Specifies that the action on a notified resource should be run before processing the resource block in which the notification is located.

:delayed
: Default. Specifies that a notification should be queued up, and then executed at the very end of the chef-client run.

:immediate, :immediately
: Specifies that a notification should be run immediately, per resource notified.

The syntax for notifies is:

notifies :action, 'resource[name]', :timer

to do this command before some command

notifies :restart, 'file[/var/www/html/index.html]', :immediate

****************************Recipe*******************************

subscribes

Ruby Type: Symbol, ‘Chef::Resource[String]’

A resource may listen to another resource, and then take action if the state of the resource being listened to changes. Specify a 'resource[name]', the :action to be taken, and then the :timer for that action.

A timer specifies the point during the chef-client run at which a notification is run. The following timers are available:

:before
    Specifies that the action on a notified resource should be run before processing the resource block in which the notification is located.
:delayed
    Default. Specifies that a notification should be queued up, and then executed at the very end of the chef-client run.
:immediate, :immediately
    Specifies that a notification should be run immediately, per resource notified.

subscribes :action, 'resource[name]', :timer

eg ::

subscribes :restart, 'file[/var/www/html/index.html]', :immediate

****************************Recipe*******************************

Arguments

The following arguments can be used with the not_if or only_if guard properties:

:user

    Specify the user that a command will run as. For example:

    not_if 'grep adam /etc/passwd', :user => 'adam'

:group

    Specify the group that a command will run as. For example:

```
not_if 'grep adam /etc/passwd', :group => 'adam'
```

:environment

Specify a Hash of environment variables to be set. For example:

```
not_if 'grep adam /etc/passwd', :environment => {
  'HOME' => '/home/adam'
}
```

:cwd

Set the current working directory before running a command. For example:

```
not_if 'grep adam passwd', :cwd => '/etc'
```

****************************Links*******************************

recipe >>
https://docs.chef.io/resource_execute.html

source-file >>
https://docs.chef.io/resource_cookbook_file.html

execute >>
https://docs.chef.io/resource_execute.html

user-creation>>
https://docs.chef.io/resource_user.html

running nodes >>
https://www.linode.com/docs/applications/chef/creating-your-first-chef-cookbook