ATTENTION:
Ecrire une recette Chef ne suffit pas.Il faut aussi la tester sur différentes plateformes pour s'assurer de son bon fonctionnement.Le mieux est de faire autant de tests que nécessaires sur des machines de développement plutôt que sur les machines en production pour éviter d'avoir de mauvaises surprises !!!

Recette Chef : installation d'un package

- fichier filezilla.rb du répertoire exemples_chef/cookbooks/filezilla/recipes

```
package 'filezilla' do
            action [:install]
end
```

- fichier gnome.rb du répertoire exemples_chef/cookbooks/gnome/recipes

```
    package 'gnome' do
         action [:install]
end
```

- fichier first_recipe.json du répertoire exemples_chef

```
   {"run_list":["recipe[filezilla::filezilla]","recipe[gnome::gnome]"]}
```

  Ce fichier first_recipe déclare l'utilisation des recettes install_filezilla.rb et install_gnome.rb précédentes. Pour l'éxécuter sur la machine cible, on utilise la commande :

```
     chef-client -j first_recipe.json
```

Recette Chef avec notion de boucle

- fichier texmaker.rb du répertoire exemples_chef/cookbooks/texmaker/recipes

```
for p in ["geany", "texmaker", "unrar", "okular-extra backends", "valgrind","check", "qpdf"] do
package p do
action [:install]
end
end
```

Recette Chef où on utilise les templates et des variables

- fichier ntp.rb du répertoire exemples_chef/cookbooks/ntp/recipes

```
    template '/etc/ntp.conf' do
    source 'ntp.erb'
            variables( :ntp_server => "ntp.mon_domaine.fr" )
            notifies :restart, "service[ntp]"
end
```

Fichier template ntp.erb dans templates/default:

Ce fichier est défini à partir d'un fichier ntp.conf standard dans lequel on définit la variable ntp_server comme suit :

```
server <%= @ntp_server %> iburst
```

Recette Chef où on utilise les templates

- fichier dns.rb du répertoire exemples_chef/cookbooks/dns/recipes

```
    template '/etc/resolv.conf' do
        source 'resolv.conf.erb'
            mode  644
            owner "root"
            group  "root"
    end
```

- fichier serveurs.rb du répertoire exemples_chef/cookbooks/dns/attributes

Ce fichier est celui dans lequel on définit les adresses IP des serveur1 (A.B.C.D) et serveur2 (E.F.G.H) qui nous servent de serveurs DNS et on le définit comme suit :

```
default['dns']['serveur1']='A.B.C.D'
default['dns']['serveur2']='E.F.G.H'
```

Fichier template resolv.conf.erb dans templates/default:

Ce fichier est défini à partir d'un fichier resolv.conf standard dans lequel on utilise les attributs default['dns']['serveur1'] et default['dns']['serveur2'] définis plus haut :

```
domain math.u-psud.fr
search math.u-psud.fr
nameserver <%= @node[:dns][:serveur1] %>
nameserver <%= @node[:dns][:serveur2] %>
```

Recette Chef : utilisation de Bash

- fichier matlab.rb du répertoire exemples_chef/cookbooks/matlab/recipes

```
if node['kernel']['machine']=='x86_64'
    remote_file '/usr/local/matlab.tgz' do
        source 'https://IP_server_source_matlab/matlab_R2016a_64b.tgz'
        owner 'root'
        group 'root'
        mode '0644'
        action :create
    end
else
    remote_file '/usr/local/matlab.tgz' do
        source 'https://IP_server_source_matlab/matlab_R2012a_32b.tgz'
        owner 'root'
        group 'root'
        mode '0644'
        action :create
    end
end


  script 'Installation de Matlab' do
    interpreter 'bash'
    user 'root'
    cwd '/tmp'
    code <<-EOH
    #!/bin/bash
    cd /usr/local/
    rm -rf MATLAB
    tar xzfp matlab.tgz
    rm matlab.tgz
EOH
end
```

Recette Chef : pour aller plus loin (installation de Chocolatey)

- fichier default.rb du répertoire cookbooks/chocolatey :

```
  unless node['platform_family'] == 'windows'
    return "Chocolatey install not supported on #{node['platform_family']}"
end
Chef::Resource.send(:include, Chocolatey::Helpers)
install_ps1 = File.join(Chef::Config['file_cache_path'], 'chocolatey-install.ps1
')
cookbook_file install_ps1 do
    action :create
    backup false
  source 'install.ps1'
end
powershell_script 'Install Chocolatey' do
    environment node  ['chocolatey']['install_vars']
    cwd Chef::Config['file_cache_path']
    code install_ps1
    not_if { chocolatey_installed? && (node['chocolatey']['upgrade'] == false) }
end
```

Recette Chef : pour aller encore plus loin - installation d'un package en fonction de la distribution

- fichier install_distrib.rb du répertoire cookbooks/installation_generique

```
  pkg_resource = case node['platform_family']
  when "debian"
        :dpkg_package
  when "fedora", "rhel", "amazon"
        :rpm_package
end

pkg_path = ( pkg_resource == :dpkg_package ) ? "/tmp/foo.deb" : "/tmp/foo.rpm"

declare_resource(pkg_resource, pkg_path) do
  action :install
end
```

- fichier install_apache_distrib.rb du répertoire cookbooks/installation_generique

```
  package "apache2" do
  case node["platform"]
  when "centos","redhat",
  "fedora","suse"

  package_name "httpd"
  when "debian","ubuntu"
    package_name "apache2"
  when "arch"
    package_name "apache"
  end
  action :install
end
```

Recette Chef : notion de rôle

- fichier cache.json du répertoire exemples_chef/roles

```
  {
"name":"cache",
"default_attributes":{
    "memcached": { "port" : 11212 }
  },
"json_class":"Chef::Role",
"env_run_lists":{},
"run_list":["recipe[memcached::memcached]","recipe[munin::munin]"],
"description":"",
"chef_type":"role",
"override_attributes":{}
}
```

Ce fichier cache.json déclare un rôle **cache**, pour lequel il faut exécuter les recettes memcached et munin, et où le port de memcached sera 11212.

Pour l'éxécuter sur la machine cible,si le répertoire /etc/chef contient le fichier **cache** on utilise la commande :

```
chef-client -j /etc/chef/cache.json
```

ATTENTION:
Pour éviter d'avoir le message **FATAL: Cannot load configuration from cache.json**, on doit lorsqu'on utilise l'option -j de la commande **chef-client** préciser le chemin absolu du fichier JSON qu'on souhaite exécuter.

- fichier lamp.json du répertoire exemples_chef/roles

```
{
  "name":"lamp",
  "default_attributes":{
  },
  "json_class":"Chef::Role",
  "env_run_lists":{},
  "run_list":["recipe[installation_generique::apache]","recipe[installation_generique::php]","recipe[installatio
n_generique::mysql]"],
  "description":"",
  "chef_type":"role",
  "override_attributes":{}
}
```

Ce fichier lamp.json déclare un rôle lamp, pour lequel il faut exécuter les recettes **apache**,**php**,**mysql** du livre de recettes **installation_generique**.

Exemple utilisant le paramètre override_attributes :

- fichier default.rb du répertoire exemples_attributs_chef/cookbooks/starter/recipes

```
log "Welcome to Chef, #{node["starter_name"]}!" do
level :info
end
```

- fichier starter.json du répertoire exemples_attributs_chef/roles

```
{
name "starter"
description "An example Chef role"
run_list "recipe[starter]"
override_attributes({
"starter_name" => "Laurent Dang",
})
}
```

Dans cet exemple,on utilise le paramètre **override_attributes** pour surcharger l'attribut starter_name défini dans le répertoire exemples_attributs_chef/cookbooks/starter/attributes/default

```
default["starter_name"] = "Sam Doe"
```

D'autres exemples de recettes sont également disponibles dans l'archive que je mets à disposition.