# Cookbooks

## Organizing Recipes





CHEF™

# Objectives

After completing this module you should be able to:

- Generate a Chef cookbook
- Use version control
- Implement the include_recipe method
- Apply a run-list of recipes to a system
- Define a Chef cookbook that sets up a web server
- Use `cookstyle` for linting cookbooks

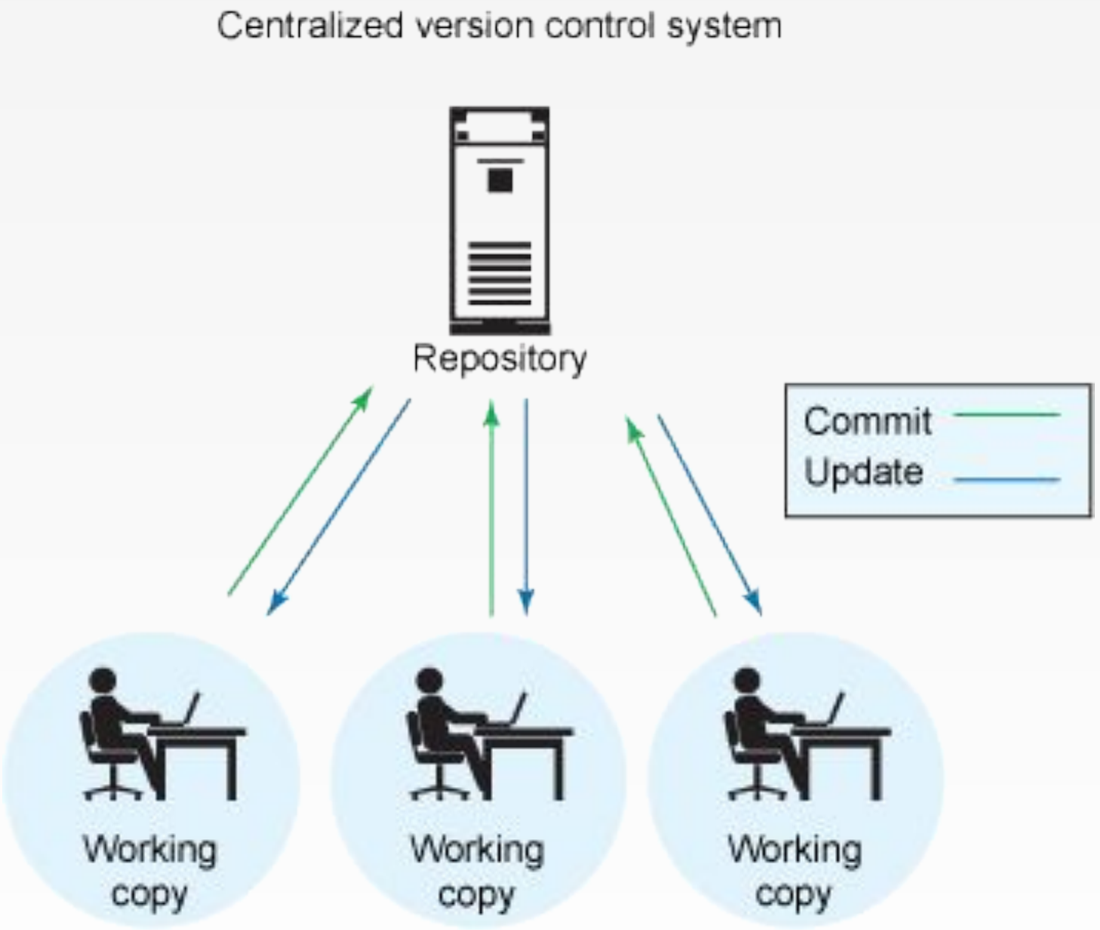
# Collaboration and Version Control

A versioning system should include:

A Central Repository into which all the developers publish their work.

Each revision should be stored as a new version.

For each change, a commit message should be added so that everyone knows what has or has not been changed.



# Versioning Pros and Cons

```
> cp hello.rb hello.rb.bak
or
> cp hello.rb hello-20170401.rb
or
> cp setup.rb setup-20170401-username.rb
```

Saving a copy of the original file as another filename.

# Git Version Control

**git** is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows.

We will be using **git** throughout this module so you can get an overview of how **git** works.

# Git Version Control

Learning everything about **git** is beyond the scope of this course. So you should try to learn more git or a version control system of your organization's choice.

There are many free resources that you can use to learn more about **git**. For example:

https://git-scm.com/doc

# GL: Setup Your Global git Configuration

```
> git config --global user.email "you@example.com"
```





CHEF

# GL: Setup Your Global git Configuration

```
> git config --global user.name "Your Name"
```





CHEF

# Cookbooks

A Chef cookbook is the fundamental unit of configuration and policy distribution.

Each cookbook defines a scenario, such as everything needed to install and configure MySQL, and then it contains all of the components that are required to support that scenario.

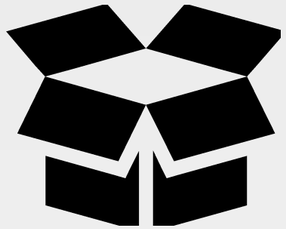
Read the first three paragraphs here: http://docs.chef.io/cookbooks.html





CHEF

CONCEPT



# Cookbook

A cookbook usually maps 1:1 to an application or to a scenario. When we define a cookbook we usually have a goal in mind that this cookbook will accomplish.

In our case we are interested in a cookbook that configures a workstation. So within that cookbook we will define all the recipes to accomplish this goal.

# GL: Setting up a Workstation

*How are we going to manage this file? Does it need a README?*

**Objective:**

- ❑ Create a cookbook
- ❑ Use git
- ❑ Copy the disable-uac recipe within the cookbook
- ❑ Use the include_recipe method to insert disable-uac recipe
- ❑ Apply the default recipe to the workstation

# GL: Create a 'cookbooks' Directory

```
> mkdir cookbooks
```

```
 Directory: C:\Users\Administrator


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----          7/12/2020   4:28 PM                cookbooks
```





CHEF

# GL: Locating the cookbooks Directory

```
> dir
```

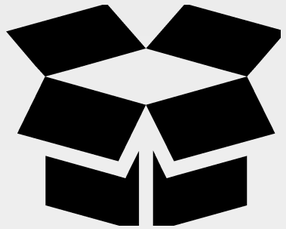
```
    Directory: C:\Users\Administrator

d----           8/3/2018  10:33 PM                .atom
d----          6/27/2020   3:54 PM                .chef
d----          6/26/2020   7:21 PM                .chef-workstation
d-r--         11/13/2017   8:03 PM                Contacts
d----          6/27/2020   4:56 PM                cookbooks
d-r--          6/26/2020   7:50 PM                Desktop
...
d----          6/27/2020   3:54 PM                nodes
d-r--         11/13/2017   8:03 PM                Pictures
...
-a---          6/27/2020   4:56 PM             48 .gitconfig
-a---          6/27/2020   4:24 PM            349 disable-uac.rb
-a---          6/27/2020   4:17 PM             47 goodbye.rb
-a---          6/26/2020   7:49 PM             56 hello.rb
-a---         11/13/2017   8:26 PM            880 kitchen-template.yml
```





CHEF



# What is 'chef'?

An executable program that allows you to generate cookbooks and cookbook components.

# GL: What Can 'chef' Do?

```
> chef --help
```

```
Usage:
    chef -h/--help
    chef -v/--version
    chef command [arguments...] [options...]

Available Commands:
    exec          Runs the command in context of the embedded ruby
    gem           Runs the `gem` command in context of the embedded ruby
    generate      Generate a new app, cookbook, or component
    shell-init    Initialize your shell to use ChefDK as your primary ruby
    install       Install cookbooks from a Policyfile and generate a locked
cookbook set
    update        Updates a Policyfile.lock.json with latest run_list and cookbooks
```

# GL: What Can 'chef generate' Do?

```
> chef generate --help
Usage: chef generate GENERATOR [options]

Available generators:
cookbook    Generate a single cookbook
  recipe       Generate a new recipe
  attribute    Generate an attributes file
  template     Generate a file template
  file         Generate a cookbook file
  lwrp         Generate a lightweight resource/provider
  repo         Generate a Chef policy repository
  policyfile   Generate a Policyfile for use with the install/push commands
(experimental)
```

# GL: What Can 'chef generate cookbook' Do?

```
> chef generate cookbook --help
```

```
Usage: chef generate cookbook NAME [options]
    -C, --copyright COPYRIGHT        Name of the copyright holder -
default...
    -m, --email EMAIL                Email address of the author -
defaults...
    -a, --generator-arg KEY=VALUE    Use to set arbitrary attribute KEY to
...
    -I, --license LICENSE            all_rights, httpd, mit, gplv2, gplv3 -
    -g GENERATOR_COOKBOOK_PATH,      Use GENERATOR_COOKBOOK_PATH for the
       --generator-cookbook
```

# GL: Let's Create a Cookbook

```
> chef generate cookbook cookbooks\workstation
+---------------------------------------------+
         Chef License Acceptance

Before you can continue, 1 product license

must be accepted. View the license at

https://www.chef.io/end-user-license-agreement/

License that need accepting:

  * Chef Workstation
Do you accept the 1 product license (yes/no)?

 > yes
Persisting 1 product license...
1 product license persisted
+---------------------------------------------+
Generating cookbook workstation
```

Be sure to accept the license by typing **yes** when prompted.





CHEF

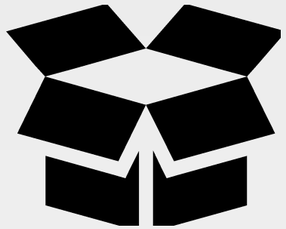
# GL: The Cookbook Has a README

```
> tree /f cookbooks\workstation
Folder PATH listing
Volume serial number is B04A-119C
C:\USERS\ADMINISTRATOR\COOKBOOKS\WORKSTATION
|   ...
|   .kitchen.yml
|   Berksfile
|   chefignore
|   metadata.rb
|   README.md
|
├───recipes
|       default.rb
|
```





CHEF



# README.md

The description of the cookbook's features written in Markdown.

http://daringfireball.net/projects/markdown/syntax

# GL: Let's Take a Look at the README

```
> gc cookbooks\workstation\README.md
```

```
# workstation

TODO: Enter the cookbook description here.
```





CHEF

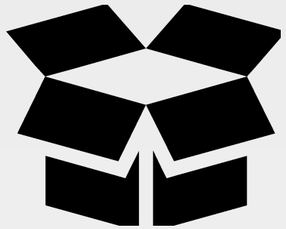
# GL: The Cookbook Has Some Metadata

```
> tree /f cookbooks\workstation
Folder PATH listing
Volume serial number is B04A-119C
C:\USERS\ADMINISTRATOR\COOKBOOKS\WORKSTATION
|   ...
|   .kitchen.yml
|   Berksfile
|   chefignore
|   metadata.rb
|   README.md
|
├───recipes
|       default.rb
|
```





CHEF



# metadata.rb

Every cookbook requires a small amount of metadata. Metadata is stored in a file called metadata.rb that lives at the top of each cookbook's directory.

http://docs.chef.io/config_rb_metadata.html

# GL: Let's Take a Look at the Metadata

```
> gc cookbooks\workstation\metadata.rb
```

```
name 'workstation'
maintainer 'The Authors'
maintainer_email 'you@example.com'
license 'All Rights Reserved'
description 'Installs/Configures workstation'
long_description 'Installs/Configures workstation'
version '0.1.0'
chef_version '>= 15.0'
```

Whenever you make a change to a cookbook, you should bump its version in that cookbook's metadata.rb file.





CHEF

# GL: The Cookbook Has a Folder for Recipes

```
> tree /f cookbooks\workstation
C:\USERS\ADMINISTRATOR\COOKBOOKS\WORKSTATION
|   kitchen.yml
|   LICENSE
|   metadata.rb
|   Policyfile.rb
|   README.md
+---.delivery
|       project.toml
+---recipes
|       default.rb
...
```





CHEF

# GL: The Cookbook Has a 'default' Recipe

```
> gc cookbooks\workstation\recipes\default.rb
```

```
# Cookbook:: workstation

# Recipe:: default

#

# Copyright:: 2020, The Authors, All Rights Reserved.
```





CHEF

# Group Exercise: Version Control

This is a probably a good point to capture the initial state of our cookbook.

- ✔ Use chef to generate a cookbook to store our disable-uac recipe.
- ❑ Add the "workstation" cookbook to version control.

# GL: Move Your Recipe into the Cookbook

```
> mv disable-uac.rb cookbooks\workstation\recipes
```





CHEF

# GL: Move to Cookbook Directory and Initialize as a git Repository

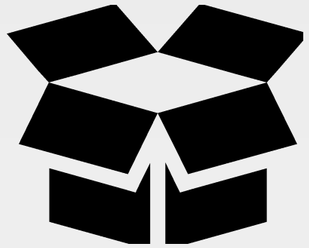
```
> cd cookbooks\workstation
> git init
```

```
Reinitialized existing Git repository in
C:/Users/Administrator/cookbooks/workstation/.git/
```





CHEF



# Staging Area

The staging area has a file, generally contained in your Git directory, that stores information about what will go into your next commit.

It’s sometimes referred to as the “index”, but it’s also common to refer to it as the staging area.

- http://git-scm.com/book/en/v2/Getting-Started-Git-Basics

# GL: Use 'git add' to Stage Files to be Committed

```
> git add .
```

```
warning: LF will be replaced by CRLF in .gitignore.

The file will have its original line endings in your working directory
```

# GL: Use 'git status'  to View the Staged Files

```
> git status
```

```
On branch master

Changes to be committed:

  (use "git reset HEAD <file>..." to unstage)



        new file:   recipes/disable-uac.rb
```





CHEF

# GL: Use 'git commit' to Save the Staged Changes

```
> git commit -m "Initial Commit"
[master f530a8f] Initial Commit
 1 file changed, 13 insertions(+)
 create mode 100644 recipes/disable-uac.rb
```



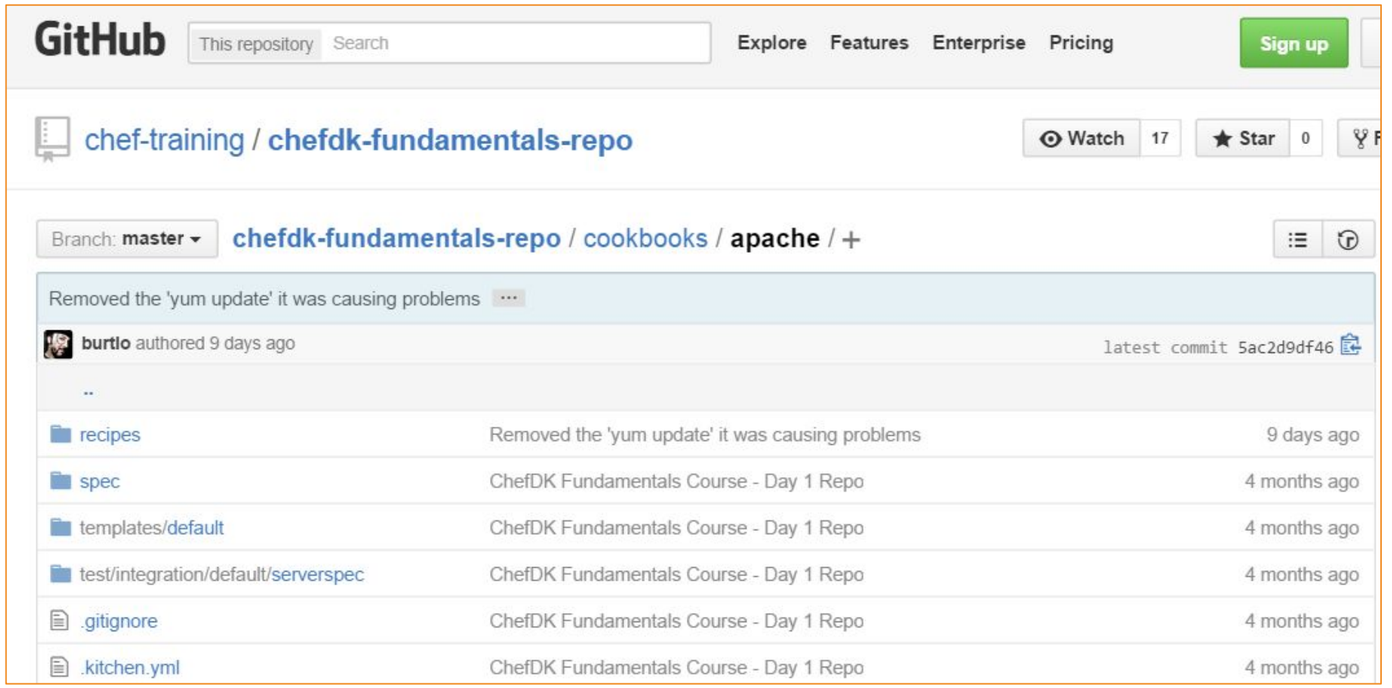


CHEF

# Git Version Control

If you use git versioning you should ultimately push the local git repository branch to a shared remote git repository.

In this way others could collaborate with you from a centralized location.







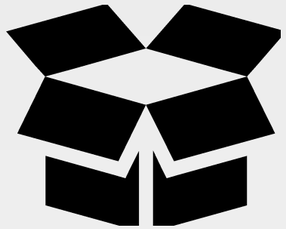

CHEF

# chef-client

```
$ chef-client --local-mode RECIPE_FILE
```

How would we apply the workstation's setup recipe?

CONCEPT

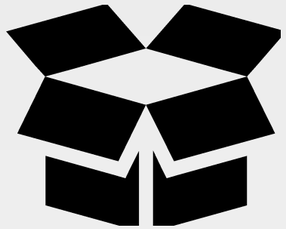


# chef-client

**chef-client** is an agent that runs locally on every node that is under management by Chef.

When a **chef-client** is run, it will perform all of the steps that are required to bring the node into the expected state.

https://docs.chef.io/chef_client.html



# --runlist "recipe[COOKBOOK::RECIPE]"

In local mode, we need to provide a list of recipes to apply to the system. This is called a **run list**. A run list is an ordered collection of recipes to execute.

Each recipe in the run list must be addressed with the format **recipe[COOKBOOK::RECIPE]**.

# GL: Applying the workstation's disable-uac recipe

```
> chef-client --local-mode --runlist "recipe[workstation::disable-uac]"
```

```
Starting Chef Infra Client, version 17.3.48
resolving cookbooks for run list: ["workstation::disable-uac"]
Synchronizing Cookbooks:
  - workstation (0.1.0)

Installing Cookbook Gems:
Compiling Cookbooks...
Converging 2 resources
Recipe: workstation::disable-uac
*
registry_key[HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System]
action create (up to date)
*
```

# GL: Setting up a Workstation

*How are we going to manage this file? Does it need a README?*

**Objective:**

- ✔ Create a cookbook
- ✔ Copy the disable-uac recipe within the cookbook
- ❑ Use the include_recipe method to insert disable-uac recipe
- ❑ Apply the default recipe to the workstation

# include_recipe method

A recipe can include one (or more) recipes located in cookbooks by using the include_recipe method. When a recipe is included, the resources found in that recipe will be inserted (in the same exact order) at the point where the include_recipe keyword is located.

https://docs.chef.io/recipes.html#include-recipes

# GL: The Default Recipe Includes the Disable Recipe

~\cookbooks\workstation\recipes\default.rb

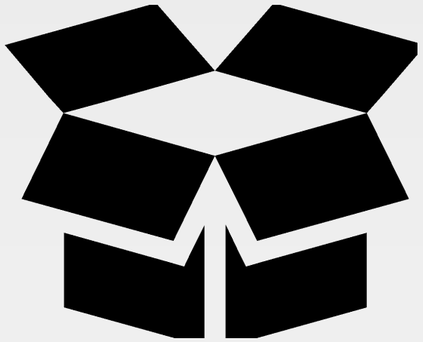
```
#
# Cookbook Name:: workstation
# Recipe:: default
#
# Copyright (c) 2017 The Authors, All Rights
Reserved.


include_recipe 'workstation::disable-uac'
```





CHEF



# -r "recipe[COOKBOOK(::default)]"

When you are referencing the default recipe within a cookbook you may optionally specify only the name of the cookbook.

chef-client understands that you mean to apply the default recipe from within that cookbook.

# GL: Apply the Cookbook's Default Recipe

```
> chef-client --local-mode -r "recipe[workstation]"
```

```
...
resolving cookbooks for run list: ["workstation"]

Synchronizing Cookbooks:

  - workstation (0.1.0)

...
Converging 2 resources

Recipe: workstation::disable-uac

  * registry_key[HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System]
action create (up to date)

  * registry_key[HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System]
action create (up to date)
```

# GL: Setting up a Workstation

*How are we going to manage this file? Does it need a README?*

**Objective:**

- ✔ Create a cookbook
- ✔ Copy the disable-uac recipe within the cookbook
- ✔ Use the include_recipe method to insert disable-uac recipe
- ✔ Apply the default recipe to the workstation

# Lab: Setting up a Web Server

- Use **`chef generate`** to create a cookbook named "**myiis**".
- Create a recipe named **`"server.rb"`** with the following policy:
  - The **powershell_script** named 'Install IIS' is run with the code: **'Add-WindowsFeature Web-Server'**
  - The **file** named **'C:\inetpub\wwwroot\Default.htm'** is created with the content:

    '<h1>Hello, world!</h1>'
  - The **service** named **'w3svc'** is enabled and started
- Use the include-recipe method to include the server recipe from within the default recipe
- Apply the default recipe and verify with **`Invoke-WebRequest localhost`**

# GL: Return to Your Home Directory

```
> cd ~
```

```
PS C:\Users\Administrator>
```





CHEF

# Lab: Create a Cookbook

```
> chef generate cookbook cookbooks\myiis
```

```
Generating cookbook myiis
- Ensuring correct cookbook content
- Committing cookbook files to git

Your cookbook is ready. Type `cd cookbooks\myiis` to enter it.

There are several commands you can run to get started locally developing and
testing your cookbook.
Type `delivery local --help` to see a full list of local testing commands.
Why not start by writing an InSpec test? Tests for the default recipe are stored
at:
test/integration/default/default_test.r

If you'd prefer to dive right in, the default recipe can be found at:
recipes/default.rb
```

# Lab: Setting up a Web Server

- ✔ Use `chef generate` to create a cookbook named "**myiis**".
- ❑ Create a recipe named `"server.rb"` with the following policy:

  The **powershell_script** named 'Install IIS' is run with the code: **'Add-WindowsFeature Web-Server'**.

  The **file** named **'C:\inetpub\wwwroot\Default.htm'** is created with the content:

  '<h1>Hello, world!</h1>'

  The **service** named **'w3svc'** is started and enabled.
- ❑ Use the include-recipe method to include the server recipe from within the default recipe
- ❑ Apply the default recipe and verify with `Invoke-WebRequest localhost`

# Lab: Create a Recipe

```
> chef generate recipe cookbooks\myiis server
...
    (diff output suppressed by config)
  * directory[cookbooks/myiis/test/integration/default] action create (up to
date)
  * template[cookbooks/myiis/test/integration/default/server_test.rb] action
create_if_missing
    - create new file cookbooks/myiis/test/integration/default/server_test.rb
    - update content in file
cookbooks/myiis/test/integration/default/server_test.rb from none to 6becfa
    (diff output suppressed by config)
  * template[cookbooks/myiis/recipes/server.rb] action create

    - create new file cookbooks/myiis/recipes/server.rb
    - update content in file cookbooks/myiis/recipes/server.rb from none to
b16ff8
...
```





CHEF

# Lab: Add Code to the Server Recipe

~\cookbooks\myiis\recipes\server.rb

```
powershell_script 'Install IIS' do
  code 'Add-WindowsFeature Web-Server'
end

file 'C:\inetpub\wwwroot\Default.htm' do
  content '<h1>Hello, world!</h1>'
end

service 'w3svc' do
  action [:enable, :start]
end
```

# Lab: Setting up a Web Server

- ✔ Use **`chef generate`** to create a cookbook named "**myiis**".
- ✔ Create a recipe named **`"server.rb"`** with the following policy:

  The **powershell_script** named 'Install IIS' is run with the code: **'Add-WindowsFeature Web-Server'**.

  The **file** named **'C:\inetpub\wwwroot\Default.htm'** is created with the content:

  '<h1>Hello, world!</h1>'

  The **service** named **'w3svc'** is started and enabled.
- ❑ Use the include-recipe method to include the server recipe from within the default recipe
- ❑ Apply the default recipe and verify with **`Invoke-WebRequest localhost`**

# Lab: The Default Recipe Includes the Disable Recipe

~\cookbooks\myiis\recipes\default.rb

```
#
# Cookbook Name:: myiis
# Recipe:: default
#
# Copyright (c) 2017 The Authors, All Rights
Reserved.


include_recipe 'myiis::server'
```

# Lab: Setting up a Web Server

- ✔ Use **`chef generate`** to create a cookbook named "**myiis**".
- ✔ Create a recipe named **`"server.rb"`** with the following policy:

  The **powershell_script** named 'Install IIS' is run with the code: **'Add-WindowsFeature Web-Server'**.

  The **file** named **'C:\inetpub\wwwroot\Default.htm'** is created with the content:

  '<h1>Hello, world!</h1>'

  The **service** named **'w3svc'** is started and enabled.
- ✔ Use the include-recipe method to include the server recipe from within the default recipe
- ❑ Apply the default recipe and verify with **`Invoke-WebRequest localhost`**

# Lab: Apply the Default Recipe

```
> chef-client -z --runlist "recipe[myiis]"
```

```
Synchronizing Cookbooks:
  - myiis (0.1.0)
Installing Cookbook Gems:
Compiling Cookbooks...
Converging 3 resources
Recipe: myiis::server
  * powershell_script[Install IIS] action run
  - execute "C:\Windows\system32\WindowsPowerShell\v1.0\powershell.exe" -NoLogo
-NonInteractive -NoProfile -ExecutionPolicy Bypass -InputFormat None -File
"C:/Users/ADMINI~1/AppData/Local/Temp/2/chef-script20200627-5836-9i1clx.ps1"

  * file[C:\inetpub\wwwroot\Default.htm] action create
    - create new file C:\inetpub\wwwroot\Default.htm
    - update content in file C:\inetpub\wwwroot\Default.htm from none to 17d291
    --- C:\inetpub\wwwroot\Default.htm  2020-06-27 18:01:15.543906000 +0000
    +++ C:\inetpub\wwwroot/chef-Default20200627-5836-f1t5x1.htm 2020-06-27
18:01:15.543906000 +0000

...
```

# Lab: Verify That the Website is Available

```
> Invoke-WebRequest localhost
```

```
StatusCode        : 200
StatusDescription : OK
Content           : <h1>Hello, world!</h1>
RawContent        : HTTP/1.1 200 OK
                    Accept-Ranges: bytes
                    Content-Length: 22
                    Content-Type: text/html
                    Date: Thu, 27 Jun 2020 18:03:25 GMT
                    ETag: "d4653250122dd51:0"
                    Last-Modified: Thu, 27 Jun 2020 18:01:15 GMT
                    Server...
Forms             : {}

Headers           : {[Accept-Ranges, bytes], [Content-Length, 22],
[Content-Type, text/html], [Date, Thu, 27 Jun 2020 18:03:25
...
```





CHEF

# Lab: Setting up a Web Server

- ✔ Use **`chef generate`** to create a cookbook named "**myiis**".
- ✔ Create a recipe named **`"server.rb"`** with the following policy:

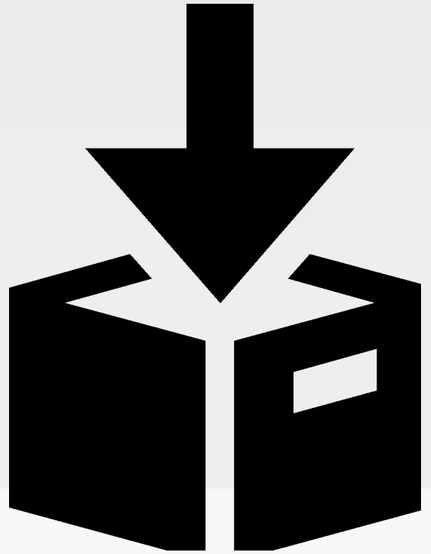
  The **powershell_script** named 'Install IIS' is run with the code: **'Add-WindowsFeature Web-Server'**.

  The **file** named **'C:\inetpub\wwwroot\Default.htm'** is created with the content:

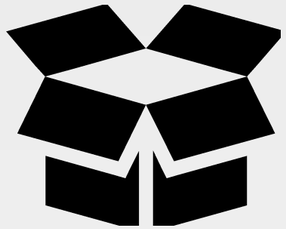
  '<h1>Hello, world!</h1>'

  The **service** named **'w3svc'** is started and enabled.
- ✔ Use the include-recipe method to include the server recipe from within the default recipe
- ✔ Apply the default recipe and verify with **`Invoke-WebRequest localhost`**

# GL: Commit Your Work



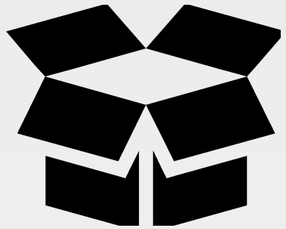
```
> cd ~\cookbooks\myiis
> git init
> git add .
> git status
> git commit -m ”Initial Commit"
```



# Committing Your Work to a Version Control Repo

When you submit or "push" your changes to a central repo like github, you should always push your changes to a github ***branch*** other than the master branch.

In this way, your changes can later be merged into the master branch in a collaborative manner, allowing for reviews of your work before merging.

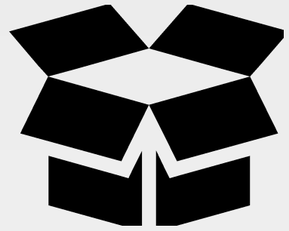


# Committing Your Work to Version Control

By now you should have an overview-level of understanding about version control and git.

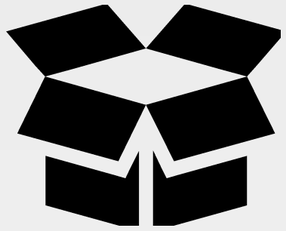
In practice you should always commit your latest changes to version control, like git.

In the interest of time we won't do any more git tasks in this course. Version control links are below in your participant guide.



# Linting with cookstyle

`cookstyle` is an easy-to-use linting tool that is included with Chef Workstation.



# Linting with cookstyle

You can simply run `cookstyle` from the directory where your recipe resides and it will indicate any syntax offenses.

https://docs.chef.io/cookstyle.html

# Example: Running cookstyle in workstation Cookbook

```
$ cookstyle
Inspecting 9 files
..C......
Offenses:

recipes/server.rb:15:1: C: 1 trailing blank lines detected.
recipes/server.rb:15:1: C: Trailing whitespace detected.

9 files inspected, 2 offenses detected
```

In this example, we ran `cookstyle` from within the recipes directory where we created the server.rb recipe. `cookstyle` has detected a trailing blank line, although it's not a critical error.

# Example: Running cookstyle in a Cookbook

```
$ cookstyle
Inspecting 6 files
..C...

Offenses:

recipes/default.rb:13:27: C: Trailing whitespace detected.
  action [:start, :enable]
                          ^
```

In this example from a different recipe, `cookstyle' even shows via the caret where the trailing whitespace exists.





CHEF

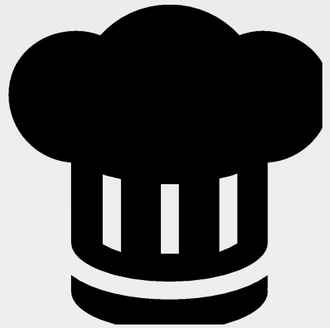
# Example: `cookstyle –a`

```
$ cookstyle -a
```

```
Inspecting 2 files
CC
Offenses:
default.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
disable-uac.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
...
2 files inspected, 10 offenses detected, 8 offenses corrected
```

`cookstyle –a` will detect and auto-correct syntax errors within the directory you are located. You may want to save a copy of the original file (or utilize git) before running `cookstyle –a` just in case you don't like the correction(s) made.

# Group Lab: cookstyle



In this brief group lab exercise you will run `cookstyle`.





CHEF

# GL: Run cookstyle at the Cookbook Level

```
> cd ~\cookbooks\workstation
> cookstyle
```

```
Inspecting 7 files
CCCC.CC
Offenses:
metadata.rb:1:1: C: Layout/EndOfLine: Carriag
name 'workstation' ...
^^^^^^^^^^^^^^^^
Policyfile.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
# Policyfile.rb - Describe how you want Chef Infra Client to build your system. ...
....
7 files inspected, 14 offenses detected recipes/default.rb:1:1: C: Layout/EndOfLine: Carriage
return character detected.
```

In this example, 7 files were inspected. Depending on how you typed your code into the cookbook's files, you should see quite a few non-critical syntax violations.

# GL: Run cookstyle at the Recipes Level

```
> cd ~\cookbooks\workstation\recipes
> cookstyle
```

```
disable-uac.rb:13:5: C: Style/HashSyntax: Use the new Ruby 1.9 hash syntax.
    :name => 'ConsentPromptBehaviorAdmin',
    ^^^^^^^^
...
disable-uac.rb:15:5: C: Style/TrailingCommaInHashLiteral: Put a comma after the last item of
a multiline hash.
    :data => 0
    ^^^^^^^^^^

2 files inspected, 10 offenses detected
```

Now you should see fewer files inspected and fewer offenses because `cookstyle` only inspected the files in the recipes directory.

# GL: Run cookstyle on an Individual File

```
> cookstyle default.rb
```

```
Inspecting 1 file

C
Offenses:

default.rb:1:1: C: Layout/EndOfLine: Carriage return character detected.
# ...
^

1 file inspected, 1 offense detected
```

While still in ~\cookbooks\workstation\recipes, run **cookstyle default.rb**
You should see only offenses in the file specified.

# Review Questions

1. What command should you use to create a cookbook?
2. What is the benefit of using include_recipe?
3. Where do you set a cookbook version?
4. What is cookstyle and from where can you get it?

# Q&A

What questions can we answer for you?

- Cookbooks
- Recipes
- Run-lists
- include_recipe method

CHEF™