LOCAL COOKBOOK DEVELOPMENT BADGE TOPICS

The Local Cookbook Development badge is awarded when someone proves that they understand the process of developing cookbooks locally. Candidates must show:

- An understanding of authoring cookbooks and setting up the local environment.
- An understanding of the Chef DK tools.
- An understanding of Test Kitchen configuration.
- An understanding of the available testing frameworks.
- An understanding of troubleshooting cookbooks.
- An understanding of search and databags.

Here is a detailed breakdown of each area.

COOKBOOK AUTHORING AND SETUP THEORY

REPO STRUCTURE - MONOLITHIC VS SINGLE COOKBOOK

Candidates should understand:
The pros and cons of a single repository per cookbook
The pros and cons of an application repository
How the Chef workflow supports monolithic vs single cookbooks
How to create a repository/workspace on the workstation

VERSIONING OF COOKBOOKS

Candidates should understand:
Why cookbooks should be versioned
The recommended methods of maintaining versions (e.g. knife spork)
How to avoid overwriting cookbooks
Where to define a cookbook version
Semantic versioning
Freezing cookbooks
Re-uploading and freezing cookbooks

STRUCTURING COOKBOOK CONTENT

Candidates should understand:
Modular content/reusability
Best practices around cookbooks that map 1:1 to a piece of software or functionality vs monolithic cookbooks
How to use common, core resources

HOW METADATA IS USED

Candidates should understand:
How to manage dependencies
Cookbook dependency version syntax
What information to include in a cookbook - author, license, etc

Metadata settings
What 'suggests' in metadata means
What 'issues_url' in metadata means

Wrapper cookbook methods

Candidates should understand:
How to consume other cookbooks in code via wrapper cookbooks
How to change cookbook behavior via wrapper cookbooks
Attribute value precedence
How to use the `include_recipe` directive
What happens if the same recipe is included multiple times
How to use the 'depends' directive

Using community cookbooks

Candidates should understand:
How to use a public and private Supermarket
How to use community cookbooks
How to wrap community cookbooks
How to fork community cookbooks
How to use Berkshelf to download cookbooks
How to configure a Berksfile
How to use a Berksfile to manage a community cookbook and a local cookbook with the same name

Using Chef Resources vs arbitrary commands

Candidates should understand:
How to shell out to run commands.
When/not to shell out
How to use the `execute` resource
When/not to use the 'execute' resource
How to ensure idempotence

CHEF DK TOOLS

'chef' command

Candidates should understand:
What the 'chef' command does
What 'chef generate' can create
How to customize content using 'generators'
The recommended way to create a template
How to add the same boilerplate text to every recipe created by a team
The 'chef gem' command

foodcritic

Candidates should understand:

What Foodcritic is
Why developers should lint their code
Foodcritic errors and how to fix them
Community coding rules & custom rules
Foodcritic commands
Foodcritic rules
How to exclude Foodcritic rules

BERKS

Candidates should understand:
How to use berks to work with upstream dependencies
How to work with GitHub & Supermarket
How to work with dependent cookbooks
How to troubleshoot berks issues
How to lock cookbook versions
How to manage dependencies using berks
berks commands

RUBOCOP

Candidates should understand:
How to use RuboCop to check Ruby styles
RuboCop vs Foodcritic
RuboCop configuration & commands
Auto correction
How to be selective about the rules you run

TEST KITCHEN

Candidates should understand:
Writing tests to verify intent
How to focus tests on critical outcomes
How to test each resource component vs how to test for desired outcomes
Regression testing

TEST KITCHEN

DRIVERS

Candidates should understand:
Test Kitchen provider & platform support
How to use .kitchen.yml to set up complex testing matrices
How to test a cookbook on multiple deployment scenarios
How to configure drivers

PROVISIONER

Candidates should understand:
The available provisioners

How to configure provisioners
When to use chef-client vs. chef-solo vs. Chef
How to use the shell provisioner

SUITES

Candidates should understand:
What a suite is
How to use suites to test different recipes in different environments
Testing directory for InSpec
How to configure suites

PLATFORMS

Candidates should understand:
How to specify platforms
Common platforms
How to locate base images
Common images and custom images

KITCHEN COMMANDS

Candidates should understand:
The basic Test Kitchen workflow
'kitchen' commands
When tests get run
How to install bussers
What 'kitchen init' does

COOKBOOK COMPONENTS

DIRECTORY STRUCTURE OF A COOKBOOK

Candidates should understand:
What the components of a cookbook are
What siblings of cookbooks in a repository are
The default recipe & attributes files
Why there is a 'default' subdirectory under 'templates'
Where tests are stored

ATTRIBUTES AND HOW THEY WORK

Candidates should understand:
What attributes are
Attributes as a nested hash
How attributes are defined
How attributes are named
How attributes are referenced
Attribute precedence levels
What Ohai is

What the 'platform' attribute is
How to use the 'platform' attribute in recipes

Files and templates - difference and how they work, when to use each

Candidates should understand:
How to instantiate files on nodes
The difference between 'file', 'cookbook_file', 'remote_file', and 'template'
How two teams can manage the same file
How to write templates
What 'partial templates' are
Common file-related resource actions and properties
ERB syntax

Custom Resources - how they are structured and where they go

Candidates should understand:
What custom resources are
How to consume resources specified in another cookbook
Naming conventions
How to test custom resources

Libraries

Candidates should understand:
What libraries are and when to use them
Where libraries are stored

AVAILALABLE TESTING FRAMEWORKS

INSPEC

Candidates should understand:
How to test common resources with InSpec
InSpec syntax
How to write InSpec tests
How to run InSpec tests
Where InSpec tests are stored

CHEFSPEC

Candidates should understand:
What ChefSpec is
The ChefSpec value proposition
What happens when you run ChefSpec
ChefSpec syntax
How to write ChefSpec tests
How to run ChefSpec tests
Where ChefSpec tests are stored

GENERIC TESTING TOPICS

Candidates should understand:
The test-driven development (TDD) workflow
Where tests are stored
How tests are organized in a cookbook
Naming conventions - how Test Kitchen finds tests
Tools to test code "at rest"
Integration testing tools
Tools to run code and test the output
When to use ChefSpec in the workflow
When to use Test Kitchen in the workflow
Testing intent
Functional vs unit testing

TROUBLESHOOTING

READING TEST-KITCHEN OUTPUT

Candidates should understand:
Test Kitchen phases and associated output

COMPILE VS. CONVERGE

Candidates should understand:
What happens during the compile phase of a chef-client run
What happens during the converge phase of a chef-client run
When pure Ruby gets executed
When Chef code gets executed

SEARCH AND DATABAGS

DATA BAGS

Candidates should understand:
What databags are
Where databags are stored
When to use databags
How to use databags
How to create a databag
How to update a databag
How to search databags
Chef Vault
The difference between databags and attributes
What 'knife' commands to use to CRUD databags

SEARCH

Candidates should understand:
What data is indexed and searchable

Why you would search in a recipe
Search criteria syntax
How to invoke a search from the command line
How to invoke a search from within a recipe