# fpm-cookery Documentation

***Release 0.32.0***

**Bernd Ahlers**

**Apr 28, 2017**

# Contents

**Current version:** 0.32.0

fpm-cookery provides an infrastructure to automatically build software based on recipes. It's heavily inspired and borrows code from the great homebrew and brew2deb projects.

CHAPTER 1

# Features

- Source archive download and caching.
- Recipes to describe and execute the software build. (e.g. configure, make, make install)
- Sandboxed builds.
- Package creation via fpm.
- Standalone recipe trees/books/you name it. No need to put the recipes into the fpm-cookery source tree.

CHAPTER 2

# Documentation Contents

## Getting Started

This page helps you to get started with the fpm-cookery tool and guides you through the installation and the creation of a simple recipe to build your first package.

You will create a package for the tmux program.

### Prerequisites

The following instructions have been tested with an Ubuntu 12.04 Linux system. It might work on other versions or other Linux systems but that cannot be guaranteed. Please use something like Vagrant to create an Ubuntu 12.04 VM if you do not have one at hand.

### Installation

#### Rubygems

fpm-cookery is written in Ruby. Before we can actually install the rubygem, you have to install a Ruby interpreter and some build tools. Execute the following to install the required packages:

```
$ sudo apt-get install ruby1.9.1 ruby1.9.1-dev build-essential curl
```

Ruby 1.9 includes the `gem` program to install rubygems:

```
$ sudo gem install fpm-cookery
```

This installs the fpm-cookery rubygem and its dependencies. At the end you should see something like "Successfully installed fpm-cookery-0.32.0".

Your fpm-cookery installation is ready to build some packages now!

### OS Package

We are planning to provide a packaged version for different operating systems. Please use the Rubygems installation method above in the meantime.

## The Recipe

The recipe is a Ruby file that contains a simple class which acts as a DSL to set the attributes of a package (like name and version) and to describe the build and installation process of a package.

You might want to create some folders to organize your recipes:

```
$ mkdir recipes
$ mkdir recipes/tmux
$ cd recipes/tmux
$ touch recipe.rb
```

The last command creates an empty recipe file. See the following snippet for the complete recipe to build a tmux package. We will go through each step afterwards. Use your text editor to add the code to the `recipe.rb` file.

```
class Tmux < FPM::Cookery::Recipe
  description 'terminal multiplexer'

  name     'tmux'
  version  '1.9a'
  homepage 'http://tmux.sourceforce.net/'
  source   'http://freefr.dl.sourceforge.net/project/tmux/tmux/tmux-1.9/tmux-1.9a.tar.
↪gz'

  build_depends 'libevent-dev', 'libncurses5-dev'
  depends       'libevent-2.0-5'

  def build
    configure :prefix => prefix
    make
  end

  def install
    make :install, 'DESTDIR' => destdir
  end
end
```

## Example Workflow

The following commands require the `recipe.rb` recipe file created above.

```
$ fpm-cook
===> Starting package creation for tmux-1.9a (ubuntu, deb)
===>
===> Verifying build_depends and depends with Puppet
===> Verifying package: libevent-dev
===> Verifying package: libevent-2.0-5
===> Missing/wrong version packages: libevent-dev
ERROR: Not running as root; please run 'sudo fpm-cook install-deps' to install␣
↪dependencies.
```

```
$ sudo fpm-cook install-deps
===> Verifying build_depends and depends with Puppet
===> Verifying package: libevent-dev
===> Verifying package: libevent-2.0-5
===> Missing/wrong version packages: libevent-dev
===> Running as root; installing missing/wrong version build_depends and depends with␣
↪Puppet
===> Installing package: libevent-dev
===> ensure changed 'purged' to 'present'
===> All dependencies installed!
```

```
$ fpm-cook
===> Starting package creation for tmux-1.9a (ubuntu, deb)
===>
===> Verifying build_depends and depends with Puppet
===> Verifying package: libevent-dev
===> Verifying package: libncurses5-dev
===> Verifying package: libevent-2.0-5
===> All build_depends and depends packages installed
===> Fetching source:
######################################################################## 100.0%
===> Building in /home/vagrant/recipes/tmux/tmp-build/tmux-1.9a
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes

[lots of output removed]

make[1]: Nothing to be done for `install-data-am'.
make[1]: Leaving directory `/home/vagrant/recipes/tmux/tmp-build/tmux-1.9a'
===> [FPM] Converting dir to deb {}
===> [FPM] No deb_installed_size set, calculating now. {}
===> [FPM] Reading template {"path":"/var/lib/gems/1.9.1/gems/fpm-1.0.2/templates/deb.
↪erb"}
===> [FPM] Creating {"path":"/tmp/package-deb-build20140308-7998-1v6uqm5/control.tar.
↪gz","from":"/tmp/package-deb-build20140308-7998-1v6uqm5/control"}
===> [FPM] Created deb package {"path":"tmux_1.9a-1_amd64.deb"}
===> Created package: /home/vagrant/recipes/tmux/pkg/tmux_1.9a-1_amd64.deb
```

```
.
|-- cache
|   `-- tmux-1.9a.tar.gz
|-- pkg
|   `-- tmux_1.9a-1_amd64.deb
|-- recipe.rb
|-- tmp-build
|   `-- tmux-1.9a
`-- tmp-dest
    `-- usr
```

```
$ dpkg -c pkg/tmux_1.9a-1_amd64.deb
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./usr/
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./usr/share/
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./usr/share/man/
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./usr/share/man/man1/
-rw-r--r-- 0/0           93888 2014-03-08 01:26 ./usr/share/man/man1/tmux.1
drwxrwxr-x 0/0               0 2014-03-08 01:26 ./usr/bin/
```

```
-rwxr-xr-x 0/0          491016 2014-03-08 01:26 ./usr/bin/tmux
```

```
$ dpkg -I pkg/tmux_1.9a-1_amd64.deb
 new debian package, version 2.0.
 size 235488 bytes: control archive= 437 bytes.
     260 bytes,    12 lines      control
     105 bytes,     2 lines      md5sums
 Package: tmux
 Version: 1.9a-1
 License: unknown
 Vendor:
 Architecture: amd64
 Maintainer: <vagrant@ubuntu1204>
 Installed-Size: 571
 Depends: libevent-2.0-5
 Section: optional
 Priority: extra
 Homepage: http://tmux.sourceforce.net/
 Description: terminal multiplexer
```

# Using Hiera

Hiera is a hierarchical key-value lookup tool from Puppet Labs that, integrated with fpm-cookery, allows you to improve your package builds by:

- Separating data from build logic,
- Selectively overriding particular recipe attributes for different platforms, software versions, etc., and
- Staying DRY by reusing data via the `hiera` and `scope` *interpolation methods*.

## Configuring Hiera

### Controlling the Lookup Hierarchy

By default, FPM-Cookery looks for Hiera data files under the `config` subdirectory of the directory containing the target recipe. You can override this through the `--data-dir` option to `fpm-cook`. You can also set the data file directory via the `datadir=` class method while defining the recipe class:

```
class FreshRecipe < FPM::Cookery::Recipe
  datadir = "/somewhere/other/than/#{File.dirname(__FILE__)}/config"
end
```

---

**Note:** Part of the recipe initialization process involves *automatically applying data* contained in the files in the current `datadir`. If you change `datadir` after the `initialize` method completes, you must call the `apply` method manually to reconfigure the recipe according to the files in the the new `datadir`.

---

When retrieving recipe data, fpm-cookery observes the following hierarchy of files under `datadir`, ordered from highest to lowest precedence:

| Path | Description |
|---|---|
| "#{recipe.platform}.yaml", "#{recipe.platform}.json" | The platform for which the recipe is being built. Corresponds to Facter's operatingsystem fact, except that all characters are lowercase. For instance, if operatingsystem is ArchLinux, recipe.platform will be archlinux. |
| "#{recipe.target}.yaml", "#{recipe.target}.json" | The target package type. Options span all package types that FPM can build, including include rpm, apk, deb, osxpkg, and others. |
| "common.yaml", "common.json" | Intended for configuration data that is common to all builds. |

You can further influence the lookup hierarchy by setting the environment variable FPM_HIERARCHY. The value should be string containing a colon-separated list of filename stems. For example:

```
$ FPM_HIERARCHY=centos:rhel:el fpm-cook package
```

prepends centos, rhel, and el to the search hierarchy, causing fpm-cookery to attempt load data from centos.yaml, rhel.yaml, el.yaml, and their .json counterparts. The final hierarchy is:

- "centos.yaml"
- "rhel.yaml"
- "el.yaml"
- "#{recipe.platform}.yaml"
- "#{recipe.target}.yaml"
- "common.yaml"

### Other Settings

You can exercise more fine-grained control by providing the path to a Hiera configuration file via the --hiera-config option. See the Hiera docs for available configuration file options.

## Hiera in Recipes

### Lookups

fpm-cookery provides the lookup class method on all classes that inherit from FPM::Cookery::Recipe, as well as an instance method of the same name. lookup takes one mandatory argument: a key to be looked up in the Hiera data files. If Hiera locates the key, lookup returns the corresponding value; otherwise lookup returns nil.

### Writing Data Files

See the Hiera data sources documentation for an overview of Hiera data sources.

---

**Note:** Please ensure that your data files use the extensions .yaml or .json, as appropriate – Hiera ignores files with any other extension.

---

You'll probably find data files most useful for defining recipe attributes. However, key-value mappings in Hiera data sources need not correspond to recipe attributes – you can store any data you like as long as it is valid YAML or JSON:

```
name: custom-package
version: '2.1.6'
some_arbitrary_data:
  - thing one
  - thing two
  - thing: three
    is_a: hash
```

*(later on...)*

```
CustomPackageRecipe.lookup('some_arbitrary_data')
  #=> ['thing one', 'thing two', {'thing' => 'three', 'is_a' => 'hash'}]
```

### Interpolation in Data Files

Within a data file, the `%{scope("...")}` method interpolates values from the following sources:

- The current recipe class
- `FPM::Cookery::Facts`
- Facter facts

The `%{hiera("...")}` method interpolates values looked up in the data files themselves.

Say you are on an `x86_64` system, and consider the following YAML data:

```
name: something-clever
version: '0.9.0'
source: 'https://www.sporkforge.net/archive/%{scope("arch")}/%{hiera("name")}-%{hiera(
↪"version")}.tar.gz'
```

`source` evaluates like so:

```
SomethingCleverRecipe.lookup('source')
  #=> 'https://www.sporkforge.net/archive/x86_64/something-clever-0.9.0.tar.gz'
```

### Symbolized Hash Keys

Ruby's YAML library automatically converts hash keys prefixed with colons into symbols. This is good to know when using Hiera to store data relevant to methods that expect symbols in their arguments – for instance, `source`.

**BAD**:

```
source:
  - 'git://gogs.myhostname.info/labyrinthm/bowie.git'
  - with: git
    tag: 'v1.1.3'
```

**GOOD**:

```
source:
  - 'git://gogs.myhostname.info/labyrinthm/bowie.git'
  - :with: git
    :tag: 'v1.1.3'
```

### Method Signatures and Unpacking Data Structures

fpm-cookery tries to Do What You Mean when dealing when loading data from Hiera, but there are some subtleties relating to method signatures that you should be aware of.

Methods that expect a single argument are the simplest case – just provide a single key-value pair:

```
name: 'myrecipe'
```

Methods that expect multiple arguments should be given as a list:

```
depends:
  - openssl-devel
  - docker-compose
```

fpm-cookery will automatically unpack the argument list with Ruby's splat (`*`) operator when invoking the method.

Methods that expect a hash should be given as a series of key-value pairs:

```
environment:
  LC_ALL: C
  SHELLOPTS: xtrace
  PAGER: cat
```

fpm-cookery will *merge* these pairs into whatever data is already assigned as the value of the attribute, rather than replacing it.

Some methods expect a heterogeneous list of arguments, `source` being the most important of these. If you want to pass options to `source` or other such methods, use the following technique:

```
source:
  - 'https://my.subversion-server.net/trunk'
  - :revision: 92834
    :externals: false
```

This translates to a Ruby `Array`:

```
['https://my.subversion-server.net/trunk', {:revision => 92834, :externals => false}]
```

For simple sources that consist only of a URL, you can do:

```
source: 'git://our.internal-git.com/foo/bar.git'
```

### Automatic Application of Hiera Data

As part of the recipe initialization process, fpm-cookery calls `lookup` to retrieve any Hiera-defined values corresponding to recipe attribute names such as `name`, `version`, and `source`. If Hiera can locate the key, fpm-cookery automatically sets the relevant attribute to the retrieved value.

Attributes defined in Hiera data files take precedence over attributes defined in `recipe.rb`:

```
--- # common.yaml
source: https://www.repourl.org/source/neato-0.2.4-7.tar.bz2
```

```
# recipe.rb
class NeatoRecipe < FPM::Cookery::Recipe
  source 'https://www.repourl.org/source/nightly/neato-nightly.tar.gz'
end
```

This results in:

```
NeatoRecipe.source #=> https://www.repourl.org/source/neato-0.2.4-7.tar.bz2
```

## Examples

See the Redis recipe for an example of fpm-cookery and Hiera in action.

CHAPTER 3

# Indices and tables

- genindex
- modindex
- search