



# Cloud Foundry





## About This Book

### License

The Little Cloud Foundry Book book is licensed under the Attribution-NonCommercial-Share-Alike 3.0 Unported license. **You should not have paid for this book.** Donations are totally fine, though :)

You are basically free to copy, distribute, modify or display the book. However, you must attribute the book to Jonathan ``Duke" Leto and Leto Labs LLC and not use it for commercial purposes. If you would like to use the content of this book under a different license feel free to contact duke@leto.net .

You can see the full text of the license at:

<http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode>

### About The Author

Jonathan ``Duke" Leto, Founder Leto Labs LLC

### Thanks, Yall

...

### Latest Version

The latest source of this book is available at:

<http://github.com/letolabs/the-little-cloud-foundry-book>.

## Introduction

Graecum est; non legitur -- ``It's Greek to me"

### What is a Platform-As-A-Service (PAAS)?

A PAAS is middleware between the low-level infrastructure (i.e. the actual CPUs, hard-drives, RAM, etc) and the high-level applications that run on top of the platform. The boundary between the platform and the application is clear, but the boundary between the infrastructure and the platform is still in flux and in active development.

### What is Cloud Foundry?

The name ``Cloud Foundry" can be used to refer to either the still beta commercial VMware product at [cloudfoundry.com](http://cloudfoundry.com) or the Cloud Foundry Open Source Project. In this book, ``Cloud Foundry" will always refer to the open source project, unless otherwise stated.

Cloud Foundry consists of a large amount of Ruby code along with a small amount of C and Shell scripts that run on top of Ubuntu LTS. The C code is used to manage Linux containers at the kernel level using a similar strategy as the Linux kernel user space program ``lxc". CF also consists of many configuration files in YAML format.

## Features

### Supported Languages

Supported languages in mainline:

- Ruby (MRE 1.8.x, MRE 1.9.x, JRuby)
- Java
- NodeJS 0.4.x, 0.6.x
- PHP 5.3, 5.4 (ActiveState, AppFog, [paas.io](http://paas.io))
- Erlang

Supported languages in forks:

- Perl 5 (ActiveState)
- .NET (IronFoundry)
- .NET (Uhuru)
- Haskell 2011.4.0 ([paas.io](http://paas.io))

## Supported Frameworks

Mainline:

- Rails
- Sinatra
- Rack (contributed recently by paas.io)
- Java Spring

Forks:

- Catalyst (ActiveState)
- Dancer (ActiveState)

## Glossary

### ACM

Access Control Manager. A general system for implementing access control features for applications.

### caldecott

A Ruby gem which allows you to ``tunnel'' into the various services attached to your application. As an example, you can use caldecott to connect to the mysql command-line of your application running in the cloud with a command similar to

```
vmc tunnel mysql-12345
```

### cloud controller

The Cloud Controller can be thought of as the maestro or orchestrator. It also includes the endpoints which vmc communicates with.

### Chef

Chef is an open-source systems integration framework to automate cloud-related tasks.

### DEA

Droplet Execution Agent. A process which manages the running application instances.

### devops

The boundary layer between development and operations is referred to as ``devops''.

**droplet**

An application, along with all dependencies, in a compressed archive file.

**Gerrit**

A code review tool based on Git. Gerrit was originally written for the Android Open Source Project at Google and allows many kinds of complex requirements and business rules to be taken into account when reviewing code.

**IAAS**

Infrastructure-as-a-service

**health**

Applications can exhibit different states of ``health" which include:

- healthy
- stopped
- flapping
- restarting

**health manager**

The subsystem of CF which monitors the health of applications and classifies them into a certain number of ``states", such as ``healthy" or ``stopped".

**IAAS**

Infrastructure-as-a-service. This is the layer below Platform-as-a-service. An example is Amazon EC2, which takes care of the ``infrastructure" of the actual physical hardware.

**lxc**

Linux ``containers" which are similar to FreeBSD ``jails". They allow many ``virtual" instances of Linux to run inside of a ``host" Linux, which is actually running on hardware. These have been available in the mainline Linux kernel since 2.6.29.

**manifest**

A Manifest is a YAML configuration file which lists various properties and metadata about the application, such as endpoint URLs, package versions and checksums.

**PAAS**

Platform-as-a-service. Examples of this are Cloud Foundry, OpenShift, Rackspace, Google App Engine and many more.

**router**

The Router takes HTTP requests and sends them to the appropriate running application instance. It usually sits behind one or more load balancers.

**stager**

The stager takes an application, which is often a directory of files on disk, and turns it into a package that can be deployed to an application instance as part of an update or initial deploy.

**Ubuntu LTS**

Long-term supported releases of Ubuntu are supported for 2 years by Canonical.

**UAA**

User Account and Authentication

**vcap**

VMware's Cloud Application Platform. This is the central Git repository which contains the Cloud Foundry codebase.

**vmc**

Stands for ``VMware Cloud" or ``VMware controller" or whatever you want it to mean, really. It is a ruby gem which is the command-line client to endpoints which implement the Cloud Foundry API.

**warden**

Warden is the security subsystem. It manages Linux containers, including their creation, destruction and monitoring. It can be thought of as a delicious layer of Ruby on top of k1xc.

**Community**

How does one get involved with Cloud Foundry? That is currently unclear. The community is eagerly awaiting a public Gerrit instance so that developers at VMware can work in the open along with outside developers. Many Github pull requests have been submitted, but most have not been dealt with. This is partly explained by the fact that it was VMware's practice to sync to their public Github mirror roughly monthly which greatly increases the likelihood of merge conflicts.

A public Gerrit instance was just announced. This will allow internal VMware CF developers and external CF developers to work together, in public, which is a huge step in the right direction for VMware. <http://reviews.cloudfoundry.org> Developers can sign in with OpenID or a Google Account and participate in the development of Cloud Foundry.

There is an ``unofficial" cloudfoundry IRC channel on Freenode at #cloudfoundry.



# Cloud Foundry



## Deploying your App to Cloud Foundry

### Installing Cloud Foundry

#### Simplest

ActiveState Stackato

```
wget blah | curl
```

#### MCF

##### From Github

```
git clone ...
```

## History of Cloud Foundry

### Case Studies

#### ql.io

node.js grumbles

#### CF multi-node using AWS

Load Balancer: AWS elastic load balancer

Routers: 2 small instances (1.7 GB of RAM each)

DEA: 2 2xlarge instances (32 GB of RAM each), 1 xlarge instance (15 GB of RAM)

CC/HM/Nats: 1 xlarge instance (15 GB of RAM)

Database: AWS relational database service

#### NTT

Contributed memcached pull request on Github, has large internal CF cloud.

### Links

<http://cloudfoundry.org>

<https://github.com/cloudfoundry>

<http://apidocs.cloudfoundry.com>