|  |  |
| --- | --- |
| **cloudenabled** | |
| **Author** | **Sushil Kumar** |
| **Documentation Version** | **V1.0** |

**Pivotal Cloud Foundry-Developer**

**Pivotal Cloud Foundry Developer - Logging, Scale, and HA**

#### Table of Contents

* [What you will learn](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#what-you-will-learn:d680e8a854a7cbad6d490c445cba2eba)
* [Exercises](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#exercises:d680e8a854a7cbad6d490c445cba2eba)
  + [Push the articulate application](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#push-the-articulate-application:d680e8a854a7cbad6d490c445cba2eba)
  + [Access articulate logs](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#access-articulate-logs:d680e8a854a7cbad6d490c445cba2eba)
    - [Questions](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#questions:d680e8a854a7cbad6d490c445cba2eba)
  + [Access articulate events](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#access-articulate-events:d680e8a854a7cbad6d490c445cba2eba)
  + [Scale articulate](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#scale-articulate:d680e8a854a7cbad6d490c445cba2eba)
    - [Scale up](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#scale-up:d680e8a854a7cbad6d490c445cba2eba)
    - [Scale out](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#scale-out:d680e8a854a7cbad6d490c445cba2eba)
    - [Questions](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#questions-1:d680e8a854a7cbad6d490c445cba2eba)
  + [High Availability](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#high-availability:d680e8a854a7cbad6d490c445cba2eba)
    - [Questions](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#questions-2:d680e8a854a7cbad6d490c445cba2eba)
* [Beyond the class](https://content.enablement.pivotal.io/pivotal-cloud-foundry-developer/logging-scale-ha/index.html#beyond-the-class:d680e8a854a7cbad6d490c445cba2eba)

# What you will learn

* How to access application logs
* How to scale an application
* How to access events
* How Pivotal Cloud Foundry handles failed application instances

# Exercises

## Push the articulate application

1) Download the **articulate** application. Copy the file to folder: ~/pivotal-cloud-foundry-developer-workshop/articulate/ (~ is shorthand for the home directory in Linux, Mac and Unix based operating systems). You will need to create this directory in your [home](https://en.wikipedia.org/wiki/Home_directory) directory.

**NOTE:** If your browser warns you about downloading this file please proceed to download it.

[Source](https://github.com/pivotal-enablement/articulate) is not required, but you may be curious how it works as you move through the labs.

2) Push the articulate application.

$ cd ~/pivotal-cloud-foundry-developer-workshop/articulate/

$ cf push articulate -p ./articulate-0.0.1-SNAPSHOT.jar -m 512M --random-route --no-start

## Access articulate logs

1) Review the documentation on [application logging](http://docs.pivotal.io/pivotalcf/devguide/deploy-apps/streaming-logs.html).

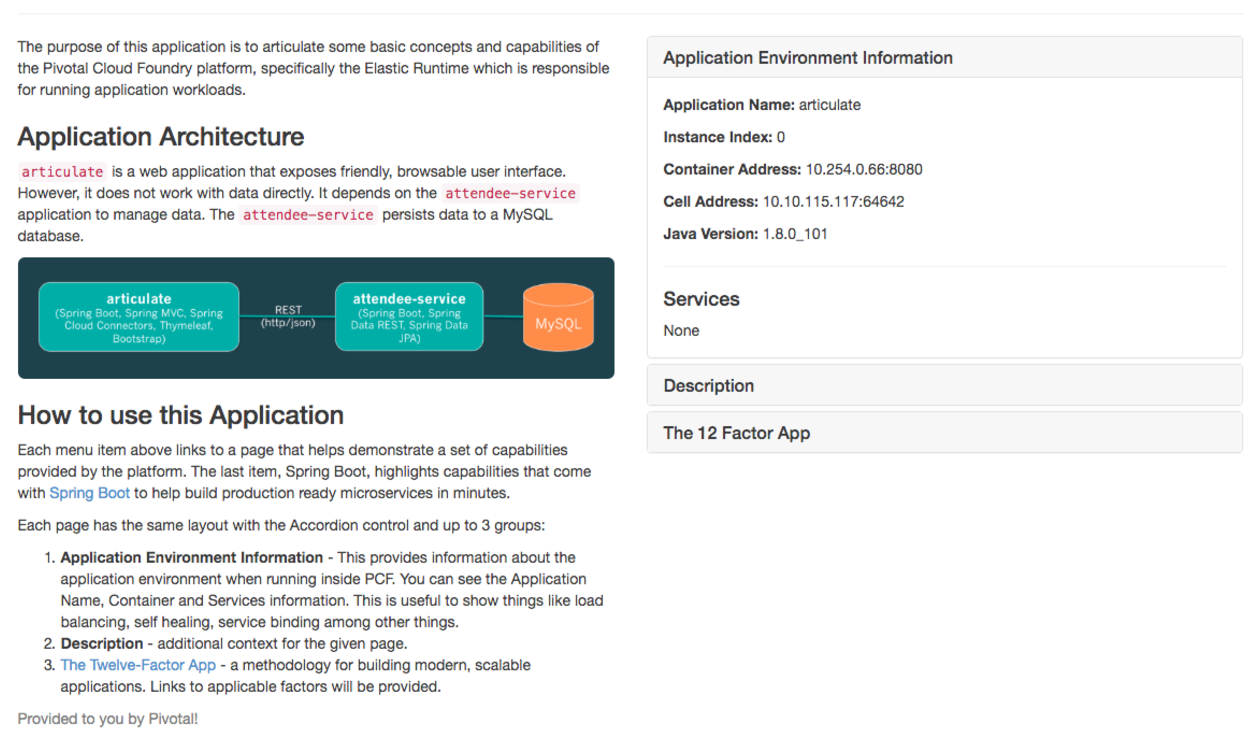
2) Tail the logs of the articulate application.

$ cf logs articulate

3) Open another terminal window and start the articulate application. Review the output from both terminal windows.

$ cf start articulate

4) Open a browser and view the articulate application. Read about our demo application.



5) Observe the log output when the articulate web page is refreshed. More logs are added!

6) Stop tailing logs

1. Go to the terminal tailing the logs
2. Send an interrupt (Control + c)

### Questions

* Where should your application write logs?
* What are some of the different origin codes seen in the log?
* How does this change how you access logs today? At scale?

## Access articulate events

Events for the application can also be used to compliment the logs in determining what has occurred with an application.

$ cf events articulate

## Scale articulate

### Scale up

1) Start tailing the logs again.

[mac, linux]

$ cf logs articulate | grep "API\|CELL"

[windows]

$ cf logs articulate | findstr "API CELL"

The above statement filters only matching log lines from the [Cloud Controller](https://docs.pivotal.io/pivotalcf/concepts/architecture/cloud-controller.html) and [Cell](https://docs.pivotal.io/pivotalcf/concepts/architecture/#diego-cell) components.

2) In another terminal window scale articulate.

$ cf scale articulate -m 1G

3) Observe log output.

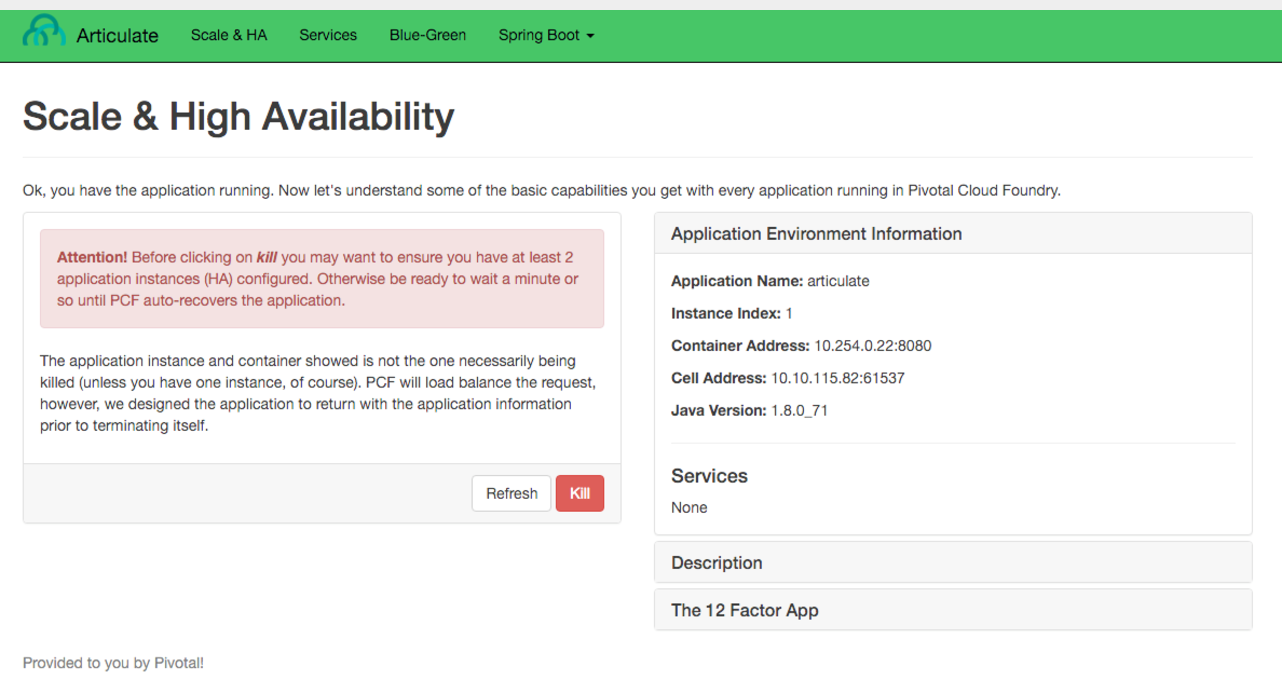
4) Stop tailing the logs.

5) Scale articulate back to our original settings.

$ cf scale articulate -m 512M

### Scale out

1) Browse to the Scale and HA page of the articulate application.



Review the Application Environment Information.

2) Press the Refresh button multiple times. All requests are going to one application instance.

3) Start tailing the logs.

[mac, linux]

$ cf logs articulate | grep 'API\|CELL'

[windows]

$ cf logs articulate | findstr "API CELL"

4) In another terminal window, scale the articulate application.

$ cf scale articulate -i 3

5) Observe log output. Then stop tailing the logs.

6) Return to articulate in a web browser. Press the Refresh button several times. Observe the Addresses and Instance Index changing.

Notice how quickly the new application instances are provisioned and subsequently load balanced!

### Questions

* What is the difference between scaling out versus scaling up?

## High Availability

Pivotal Cloud Foundry has [4 levels of HA](https://blog.pivotal.io/pivotal-cloud-foundry/products/the-four-levels-of-ha-in-pivotal-cf) (High Availability) that keep your applications and the underlying platform running. In this section, we will demonstrate one of them. Failed application instances will be recovered.

1) At this time you should be running multiple instances of articulate. Confirm this with the following command:

$ cf app articulate

2) Return to articulate in a web browser (Scale and HA page). Press the Refresh button. Confirm the application is running.

3) Kill the app. Press the Kill button!

4) Check the state of the app through the cf CLI.

$ cf app articulate

Sample output below (notice the requested state vs actual state). In this case, Pivotal Cloud Foundry had already detected the failure and is starting a new instance.

requested state: started

instances: 3/3

usage: 512M x 3 instances

urls: articulate.pcfi1.fe.gopivotal.com

last uploaded: Mon Mar 21 20:27:57 UTC 2016

stack: cflinuxfs2

buildpack: java-buildpack=v3.5.1-offline-http://github.com/pivotal-cf/pcf-java-buildpack.git#d6c19f8 java-main open-jdk-like-jre=1.8.0\_65 open-jdk-like-memory-calculator=2.0.1\_RELEASE spring-auto-reconfiguration=1.10.0\_RELEASE

state since cpu memory disk details

#0 starting 2016-03-21 04:16:23 PM 0.0% 692K **of** 512M 93.4M **of** 1G

#1 running 2016-03-21 03:28:58 PM 0.5% 410.4M **of** 512M 158.8M **of** 1G

#2 running 2016-03-21 04:15:57 PM 23.9% 357.8M **of** 512M 158.8M **of** 1G

Repeat this command as necessary until state = running.

5) In your browser, Refresh the articulate application.

The app is back up!

A new, healthy app instance has been automatically provisioned to replace the failing one.

6) View which instance was killed.

$ cf events articulate

7) Scale articulate back to our original settings.

$ cf scale articulate -i 1

### Questions

* How do you recover failing application instances today?
* What effect does this have on your application design?
* How could you determine if your application has been crashing?