CLOUD FOUNDRY

Pivotal Cloud Foundry Developer

Lab Instructions

Application Deployment using Pivotal Cloud Foundry

Version 1.11.a

Pivotal

Copyright Notice

Table of Contents

Chapter 1. Logging, Scale, and HA

Estimated Time: 30 minutes

1.1. Preface

In this lab, we build on the foundation we've picked up from the first lab. We introduce new commands that are essential for developers working with cloudfoundry. You'll learn how to tail logs, view application events, and scale an application with ease. The last portion of the lab focuses on demonstrating one of the high availability characteristics of the cloud foundry platform: automatic restart of a failed application instance. Enjoy.

1.2. Exercises

1.2.1. Push the `articulate` application

1. The `articulate` application, `articulate-0.2.jar`, should be in your lab files in its own directory. If so, jump to step (3)

2. Alternatively, you can Download the `articulate` JAR, create `pivotal-cloud-foundry-developer-workshop/articulate/` directory and copy the JAR there.

   Note

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

3. The Source is not required, but you may be curious how it works as you move through the labs.

4. Push the `articulate` application.

```
$> cd .../pivotal-cloud-foundry-developer-workshop/articulate/

$> cf push articulate -p ./articulate-0.2.jar -m 768M --random-route --no-start
```

   Notice how this time, we use the `-p` flag to tell the `cf` command what file to upload.

1.2.2. Access `articulate` logs

Before continuing this lab, review the documentation on application logging. This details the different types of messages you will see in the logs (otherwise you won't understand what you are seeing).

Note

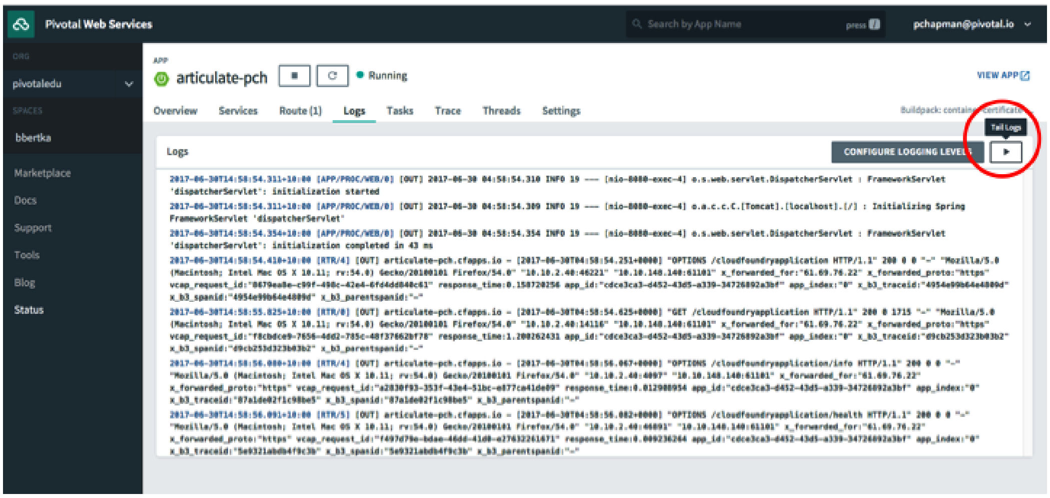
Viewing logs in real-time is termed "tailing" (a Unix term, named after the Unix `tail` command).

1. Tail the logs of the `articulate` application.

```
cf logs articulate
```

Note

Tailing the logs requires an *incoming* Websocket connection. If using PWS, this will probably be blocked by your company's firewall. If `cf logs` fails to run, go to the App Manager in your browser, open the articulate application and view the logs in the Logs tab. On the top-right there is an icon with a "Go" arrow in (which pops up `tail logs` when you mouse over it - see below). Click to start tailing mode. Once log-tailing is enabled the "Go" arrow changes to a "Pause" icon.



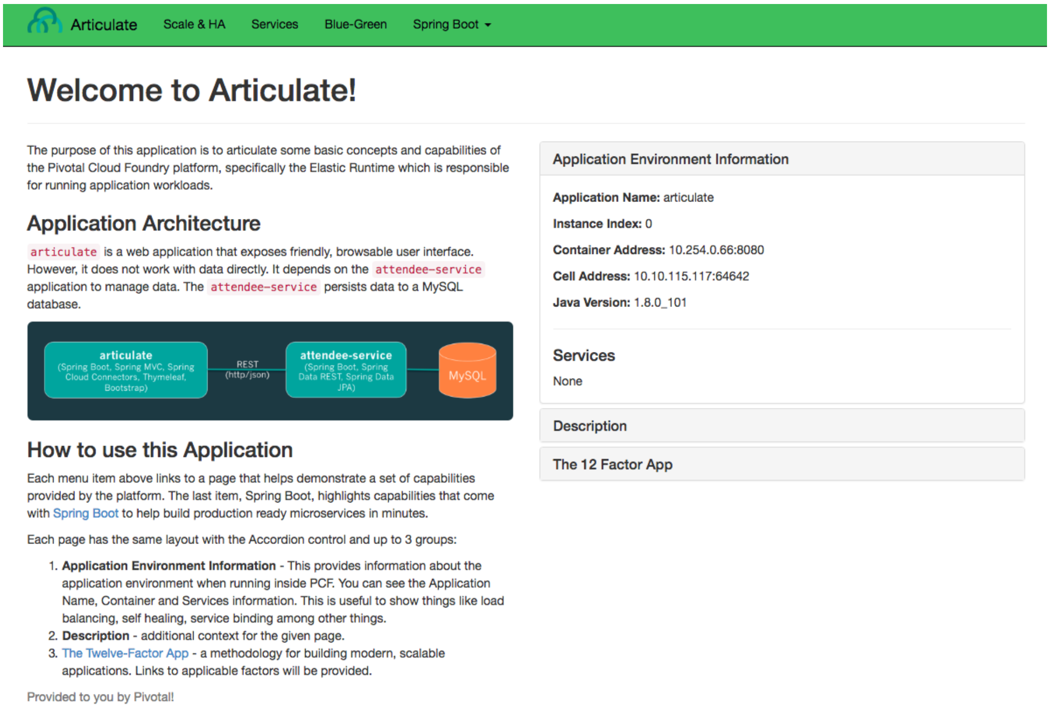
```
2017-06-30T14:58:54.311+10:00 [APP/PROC/WEB/0] [OUT] 2017-06-30 04:58:54.310 INFO 19 --- [nio-8080-exec-4] o.s.web.servlet.DispatcherServlet : FrameworkServlet
'dispatcherServlet': initialization started
2017-06-30T14:58:54.311+10:00 [APP/PROC/WEB/0] [OUT] 2017-06-30 04:58:54.309 INFO 19 --- [nio-8080-exec-4] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring
FrameworkServlet 'dispatcherServlet'
2017-06-30T14:58:54.354+10:00 [APP/PROC/WEB/0] [OUT] 2017-06-30 04:58:54.354 INFO 19 --- [nio-8080-exec-4] o.s.web.servlet.DispatcherServlet : FrameworkServlet
'dispatcherServlet': initialization completed in 43 ms
2017-06-30T14:58:54.410+10:00 [RTR/4] [OUT] articulate-pch.cfapps.io - [2017-06-30T04:58:54.251+0000] "OPTIONS /cloudfoundryapplication HTTP/1.1" 200 0 0 "-" "Mozilla/5.0
(Macintosh; Intel Mac OS X 10.11; rv:54.0) Gecko/20100101 Firefox/54.0" "10.10.2.40:46221" "10.10.148.140:61101" x_forwarded_for:"61.69.76.22" x_forwarded_proto:"https"
vcap_request_id:"0679ea0e-c99f-498c-42e4-6fd4dd848c61" response_time:0.158720256 app_id:"cdce3ca3-d452-43d5-a339-34726892a3bf" app_index:"0" x_b3_traceid:"4954e99b64e4809d"
x_b3_spanid:"4954e99b64e4809d" x_b3_parentspanid:"-"
2017-06-30T14:58:55.825+10:00 [RTR/0] [OUT] articulate-pch.cfapps.io - [2017-06-30T04:58:54.625+0000] "GET /cloudfoundryapplication HTTP/1.1" 200 0 1715 "-" "Mozilla/5.0
(Macintosh; Intel Mac OS X 10.11; rv:54.0) Gecko/20100101 Firefox/54.0" "10.10.2.40:14116" "10.10.148.140:61101" x_forwarded_for:"61.69.76.22" x_forwarded_proto:"https"
vcap_request_id:"f8cbdce9-7656-4dd2-785c-40f37662bf78" response_time:1.200262431 app_id:"cdce3ca3-d452-43d5-a339-34726892a3bf" app_index:"0" x_b3_traceid:"d9cb253d323b03b2"
x_b3_spanid:"d9cb253d323b03b2" x_b3_parentspanid:"-"
2017-06-30T14:58:56.080+10:00 [RTR/4] [OUT] articulate-pch.cfapps.io - [2017-06-30T04:58:56.067+0000] "OPTIONS /cloudfoundryapplication/info HTTP/1.1" 200 0 0 "-"
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:54.0) Gecko/20100101 Firefox/54.0" "10.10.2.40:4097" "10.10.148.140:61101" x_forwarded_for:"61.69.76.22"
x_forwarded_proto:"https" vcap_request_id:"a2830f93-353f-43e4-51bc-e877ca41de09" response_time:0.012908954 app_id:"cdce3ca3-d452-43d5-a339-34726892a3bf" app_index:"0"
x_b3_traceid:"87a1de02f1c98be5" x_b3_spanid:"87a1de02f1c98be5" x_b3_parentspanid:"-"
2017-06-30T14:58:56.091+10:00 [RTR/5] [OUT] articulate-pch.cfapps.io - [2017-06-30T04:58:56.082+0000] "OPTIONS /cloudfoundryapplication/health HTTP/1.1" 200 0 0 "-"
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:54.0) Gecko/20100101 Firefox/54.0" "10.10.2.40:46891" "10.10.148.140:61101" x_forwarded_for:"61.69.76.22"
x_forwarded_proto:"https" vcap_request_id:"f497d79e-bdae-46dd-41d8-e27632261671" response_time:0.009236264 app_id:"cdce3ca3-d452-43d5-a339-34726892a3bf" app_index:"0"
x_b3_traceid:"5e9321abdb4f9c3b" x_b3_spanid:"5e9321abdb4f9c3b" x_b3_parentspanid:"-"
```

2. Open *another* terminal window and start the `articulate` application. Compare the output from the start

command with the logging output.

```
cf start articulate
```

3. Open a browser and view the `articulate` application. Read about our demo application.



Welcome to Articulate!

The purpose of this application is to articulate some basic concepts and capabilities of the Pivotal Cloud Foundry platform, specifically the Elastic Runtime which is responsible for running application workloads.

Application Architecture

`articulate` is a web application that exposes friendly, browsable user interface. However, it does not work with data directly. It depends on the `attendee-service` application to manage data. The `attendee-service` persists data to a MySQL database.

articulate
(Spring Boot, Spring MVC, Spring Cloud Connectors, Thymeleaf, Bootstrap)
REST (http/json)
attendee-service
(Spring Boot, Spring Data REST, Spring Data JPA)
MySQL

How to use this Application

Each menu item above links to a page that helps demonstrate a set of capabilities provided by the platform. The last item, Spring Boot, highlights capabilities that come with Spring Boot to help build production ready microservices in minutes.

Each page has the same layout with the Accordion control and up to 3 groups:

1. **Application Environment Information** - This provides information about the application environment when running inside PCF. You can see the Application Name, Container and Services information. This is useful to show things like load balancing, self healing, service binding among other things.
2. **Description** - additional context for the given page.
3. The Twelve-Factor App - a methodology for building modern, scalable applications. Links to applicable factors will be provided.

Provided to you by Pivotal!

Application Environment Information

Application Name: articulate
Instance Index: 0
Container Address: 10.254.0.66:8080
Cell Address: 10.10.115.117:64642
Java Version: 1.8.0_101

Services
None

Description

The 12 Factor App

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

5. Stop tailing logs

   Do ONE of the following:

   a. **Command Line**

      Go to the terminal tailing the logs (where you ran `cf logs`) and send an interrupt (kbd:[Ctrl+C])

   b. **App Manager**

Click the "Pause" button at top-right of the logging tab.

1.2.2.1. 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?

1.2.3. Access `articulate` events

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

```
cf events articulate
```

Events can also be seen in the Overview tab when using the App Manager.

1.2.4. Scale `articulate`

Let’s scale the application up and down again and watch this happening in the logs.

Note

If viewing logs in your browser, you will have to search manually for log lines containing `API` (Cloud Controller logs) and `CELL` (Diego Cell logs).

1.2.4.1. Scale up

1. Start tailing the logs again.

   Do ONE of the following:

   a. **mac, linux, powershell**

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

We are using `grep` to only show matching log lines from the Cloud Controller and Diego Cell components.

b. **windows**

```
cf logs articulate | findstr "API CELL"
```

We are using `findstr` to only show matching log lines from the Cloud Controller and Diego Cell components.

c. **App Manager**

View `articulate` in the App Manager in your browser and enable log-tailing as described above.

You will need to search for `API` and `CELL` log messages manually.

2. In *another* terminal window scale `articulate`.

```
cf scale articulate -m 1G
```

3. Observe log output. You should see something like this:

```
2017-06-30T17:35:40.958+10:00 [API/7] [OUT] Updated app with guid cdce3ca3-d452-43d5-a339-34726892a3bf ({"memory"=>900})
2017-06-30T17:35:41.945+10:00 [API/7] [OUT] Updated app with guid cdce3ca3-d452-43d5-a339-34726892a3bf ({"state"=>"STOPPED"})
2017-06-30T17:35:44.190+10:00 [API/6] [OUT] Updated app with guid cdce3ca3-d452-43d5-a339-34726892a3bf ({"state"=>"STARTED"})
2017-06-30T17:35:44.717+10:00 [CELL/0] [OUT] Creating container
2017-06-30T17:35:45.299+10:00 [CELL/0] [OUT] Successfully created container
2017-06-30T17:35:48.441+10:00 [CELL/0] [OUT] Starting health monitoring of container
```

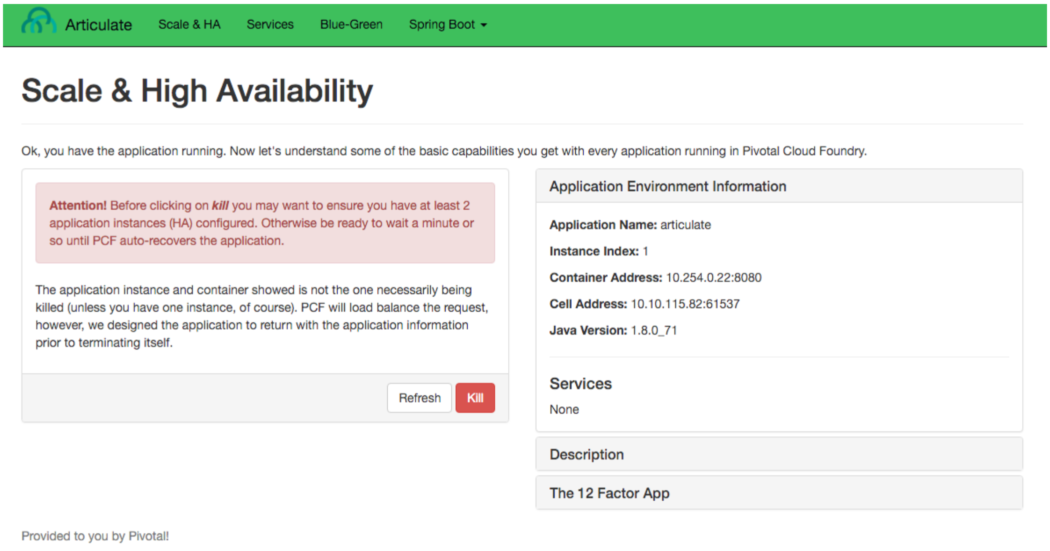
4. Stop tailing the logs.

5. Scale `articulate` back to our original settings.

```
cf scale articulate -m 768M
```

1.2.4.2. Scale out

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



Scale & High Availability

Ok, you have the application running. Now let's understand some of the basic capabilities you get with every application running in Pivotal Cloud Foundry.

Attention! Before clicking on ***kill*** you may want to ensure you have at least 2 application instances (HA) configured. Otherwise be ready to wait a minute or so until PCF auto-recovers the application.

The application instance and container showed is not the one necessarily being killed (unless you have one instance, of course). PCF will load balance the request, however, we designed the application to return with the application information prior to terminating itself.

Application Environment Information

Application Name: articulate
Instance Index: 1
Container Address: 10.254.0.22:8080
Cell Address: 10.10.115.82:61537
Java Version: 1.8.0_71

Services

None

Description

The 12 Factor App

Provided to you by Pivotal!

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 again.

   Do ONE of the following:

   1 **mac, linux, powershell**

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

   2 **windows**

```
cf logs articulate | findstr "API CELL"
```

   3 **App Manager**

   View `articulate` in the App Manager in your browser and enable log-tailing as described above.

   You will need to search for `API` and `CELL` log messages manually.

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!

1.2.4.3. Questions

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

1.2.5. High Availability

Pivotal Cloud Foundry has 4 levels of HA (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: 768M 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 768M    93.4M of 1G
```

```
#1   running    2016-03-21 03:28:58 PM   0.5%    410.4M of 768M   158.8M of 1G
#2   running    2016-03-21 04:15:57 PM   23.9%   357.8M of 768M   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
```

1.2.5.1. 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?

1.3. Beyond the class

- Try the same exercises, but using Apps Manager instead
- Visit the cloud foundry plugins site at https://plugins.cloudfoundry.org/ and investigate the *Open* plugin.