Pivotal

Pivotal Cloud Foundry

2.5 hr Crash Course

SECTION 1 PRE-REQUISITES & INTRO

Pre-Requisites

1. Download sample apps
   (Check your email)

2. Install the PCF Command Line Interface
(CLI is available at
https://apps.paas.mia.ulti.io/tools)

3. Make sure you can Login with your credentials

```
$ cf  login   -a https://api.paas.mia.ulti.io   -u user#   -p user#
```

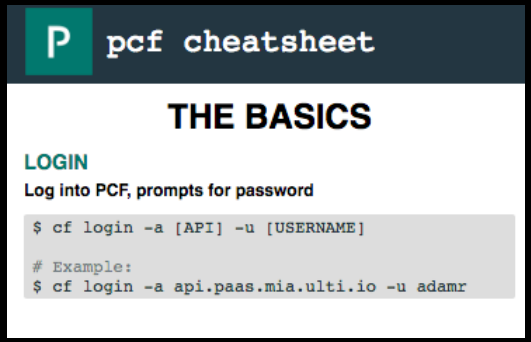
Pivotal

What to expect….

- 2.5 hours of theory and practice
- 9 Sections :
  1. Intro
  2. Login
  3. Push
  4. Services
  5. Environment vars
  6. Scaling
  7. Monitoring
  8. Languages
  9. Help links

- Guide

- Cheatsheet



Add Questions to SLACK:
#48hrs-pcf

Pivotal

Cloud Foundry….

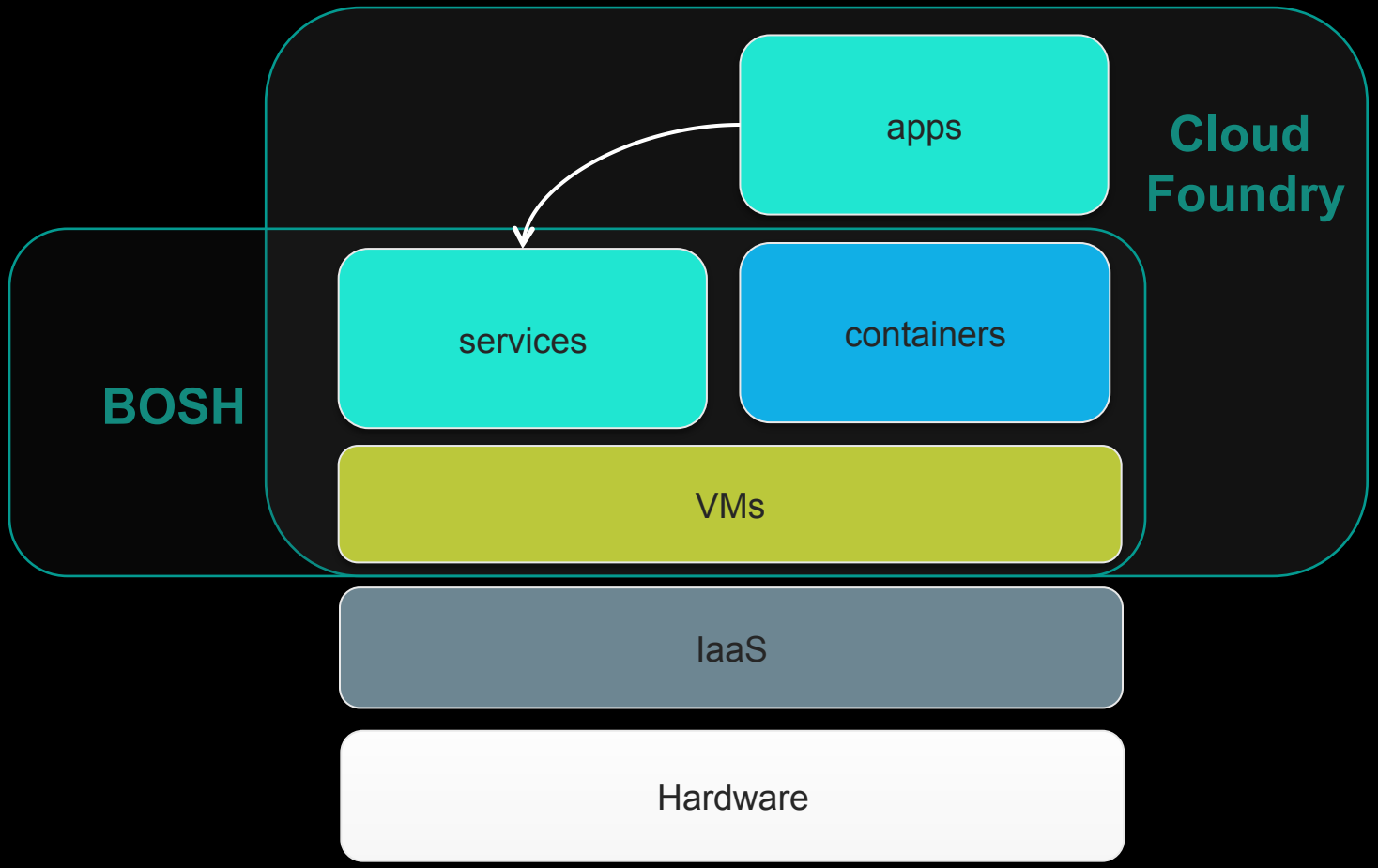
- Open Source Cloud Computing Platform

- Developed at VMware
  - Now owned by Pivotal Software
  - Pivotal is a Joint Venture of EMC, Vmware, GE, Msft, and Ford

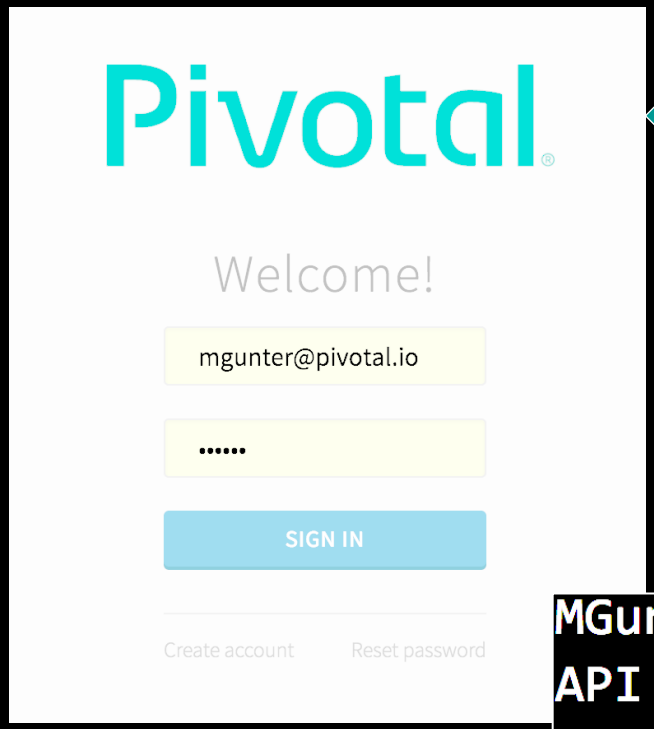
- Designed for
  - Fast application development and deployment.
  - Highly scalable and available architecture.
  - DevOps-friendly workflows.
  - Reduced chance of human error.
  - Multi-tenant compute efficiencies.
  - High degree of Operational support via autonomous resource mgmt.

Pivotal

How does it work?



SECTION 2 CF LOGIN & CF TARGET



Apps Manager

CLI
cf login....

```
MGunter-MB-Pro:crashcourse mgunter$ cf login
API endpoint: https://api.run.pivotal.io

Email> mgunter@pivotal.io

Password>
Authenticating...
OK
```

Pivotal

```
> cf help
NAME:
   cf - A command line tool to interact with Cloud Foundry

USAGE:
   cf [global options] command [arguments...] [command options]

VERSION:
   6.20.0+25b1961-2016-06-29

GETTING STARTED:
   help                                Show help
   version                             Print the version
   login                               Log user in
   logout                              Log user out
   passwd                              Change user password
   target                              Set or view the targeted org or space
```

List Commands

```
> cf apps -h
NAME:
   apps - List all apps in the target space

USAGE:
   cf apps

ALIAS:
   a
```

Show help for a single command

Pivotal

Session 2:

```
$ cf  login –a https://api.paas.mia.ulti.io
$ cf  target
API endpoint:   https://api.paas.mia.ulti.io (API version: 2.54.0)
User:           training1
Org:            48hours
Space:          training1

$ cf  - -help [optional]
```

SECTION 3 CF PUSH & RUN

CF PUSH

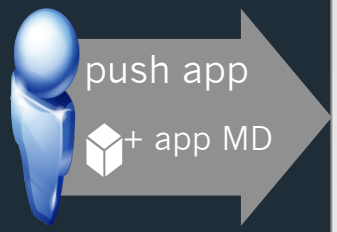
① Upload app bits and metadata

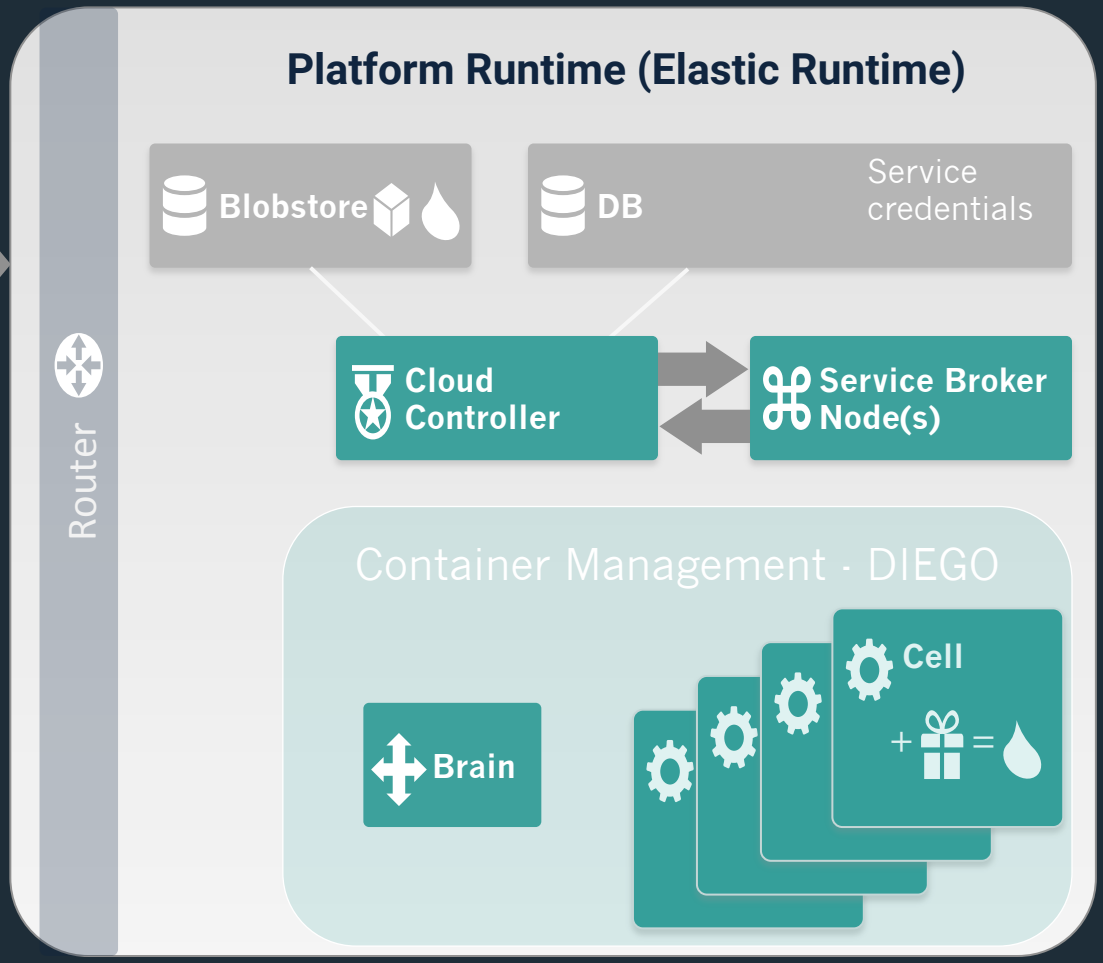
② Create and bind services

③ Stage application

④ Deploy application

⑤ Manage application health





Pivotal

https://12factor.net

The Twelve Factors

I. Codebase
One codebase tracked in revision control, many deploys

II. Dependencies
Explicitly declare and isolate dependencies

III. Config
Store config in the environment

IV. Backing services
Treat backing services as attached resources

V. Build, release, run
Strictly separate build and run stages

VI. Processes
Execute the app as one or more stateless processes

VII. Port binding
Export services via port binding

VIII. Concurrency
Scale out via the process model

IX. Disposability
Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity
Keep development, staging, and production as similar as possible

XI. Logs
Treat logs as event streams

XII. Admin processes
Run admin/management tasks as one-off processes

Pivotal

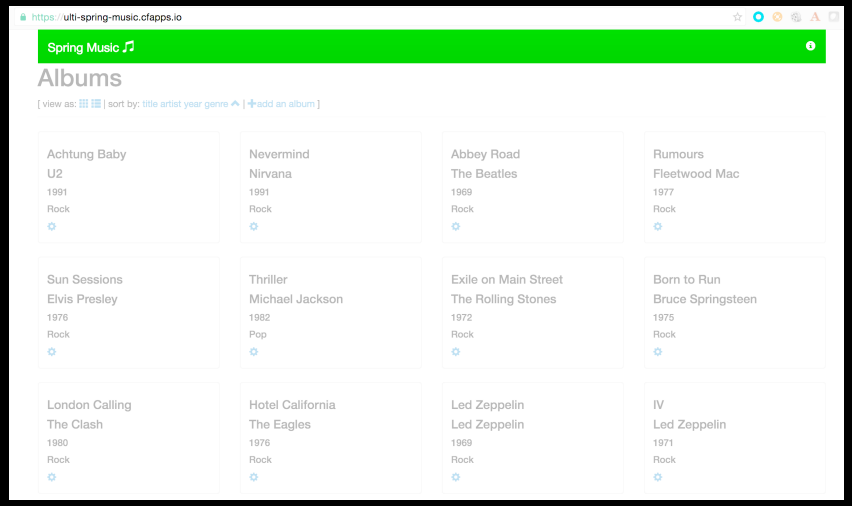
Manifest

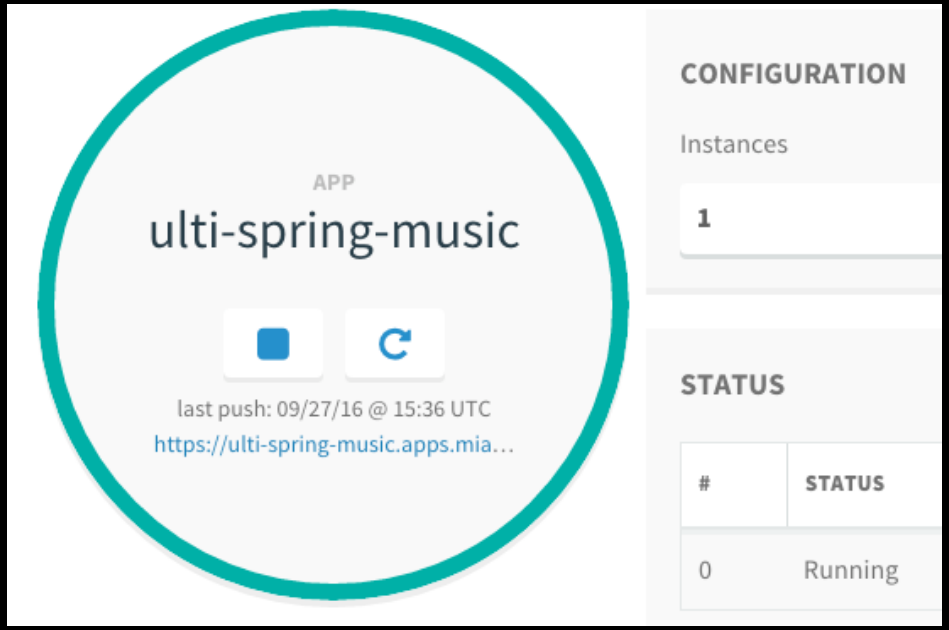
```yaml
---
instances: 1
memory: 1024M
applications:
- name: fortune-service
  host: fortunes-mg
  path: fortune-teller-fortune-service/target/fortune-teller-fort
  services:
  - fortunes-db
  - config-server
  - service-registry
- name: fortune-ui
  host: fortunes-ui-mg
  path: fortune-teller-ui/target/fortune-teller-ui-0.0.1-SNAPSHOT.jar
  services:
  - config-server
  - service-registry
  - circuit-breaker-dashboard
env:
  SPRING_PROFILES_ACTIVE: pcf
  CF_TARGET: https://api.run.pivotal.io
```

manifest.yml

- ✓ Instances
- ✓ Memory
- ✓ Multiple apps
- ✓ Names, hosts, path
- ✓ Services
- ✓ Environment vars

Pivotal





Pivotal

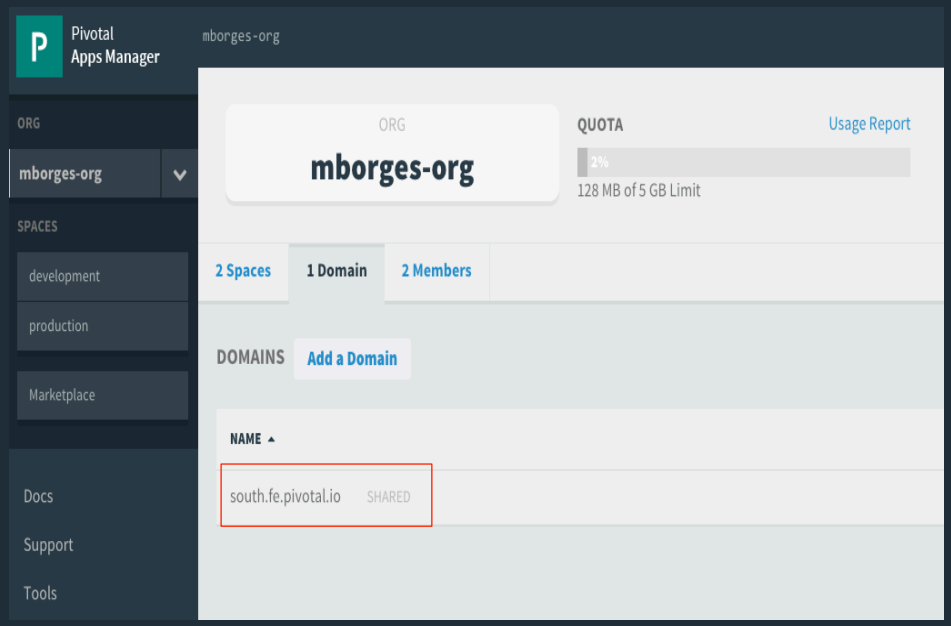
iii. Always Provide an Application Name to cf push
iv. How cf push Finds the Application
v. Precedence Between Manifests, Command Line Options, and Most Recent Values
vi. Optional Attributes
    - The buildpack attribute
    - The command attribute
    - The disk quota attribute
    - The domain attribute
    - The domains attribute
    - The stack attribute
    - The instances attribute
    - The memory attribute
    - The health-check-type attribute
    - The host attribute
    - The hosts attribute
    - The no-hostname attribute
    - The routes attribute
    - The random-route attribute
    - The path attribute
    - The timeout attribute
    - The no-route attribute
    - Environment Variables
    - Services
vii. Describing Multiple Applications with One Manifest
viii. Minimizing Duplication
ix. Multiple Manifests with Inheritance

Documentation on Manifest.yml

Pivotal

Domains

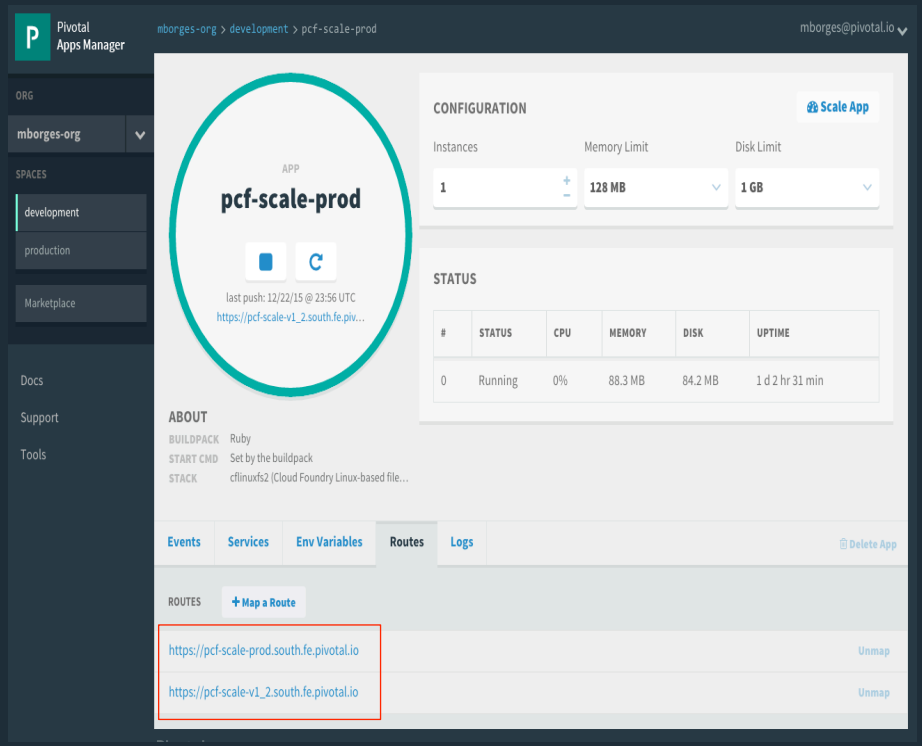
- Each Cloud Foundry installation has a default *app domain*
- Domains provide a namespace from which to create routes
- Requests for any routes created from the domain will be routed to Elastic Runtime.
- Domains can be shared or private in regards to PCF organizations



Pivotal

Routes

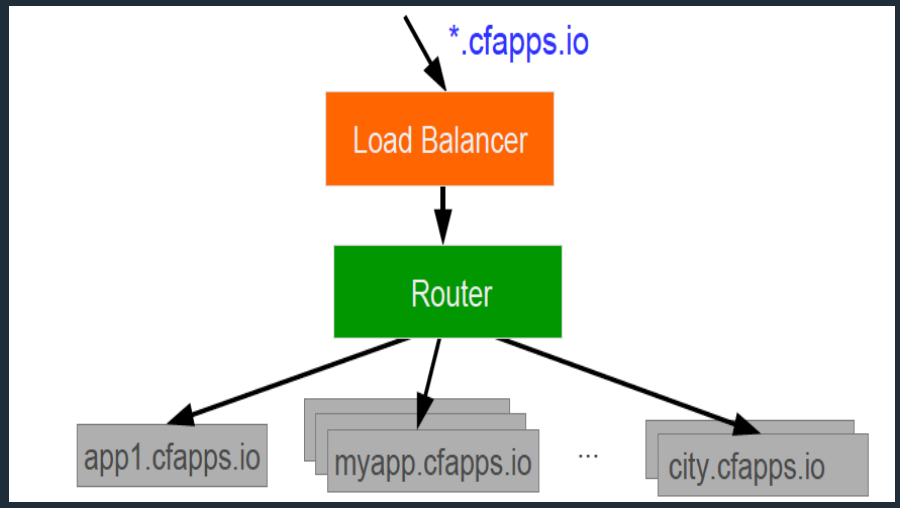
- HTTP requests are routed to apps pushed by a “ROUTE”
- Many Apps ….to a Route
- Many Routes…. to an App
- Routes belong to a space



Pivotal

Routes and Domains

- A wildcard entry (*) is added to the DNS for the app domain
- The Router uses the subdomain to map to application instance(s)
- Host and Domain are in a shared namespace.
- No duplicates allowed!



Pivotal

Session 3:

```
$ ls
spring-music.war
$ cf  push  [name]-spring-music –p spring-music.war
```

[browse to url created for app.]

```
$ cf  map-route [name]-spring-music apps.mia.ulti.io
  --hostname [new-name]
```

[browse to url created for route.]

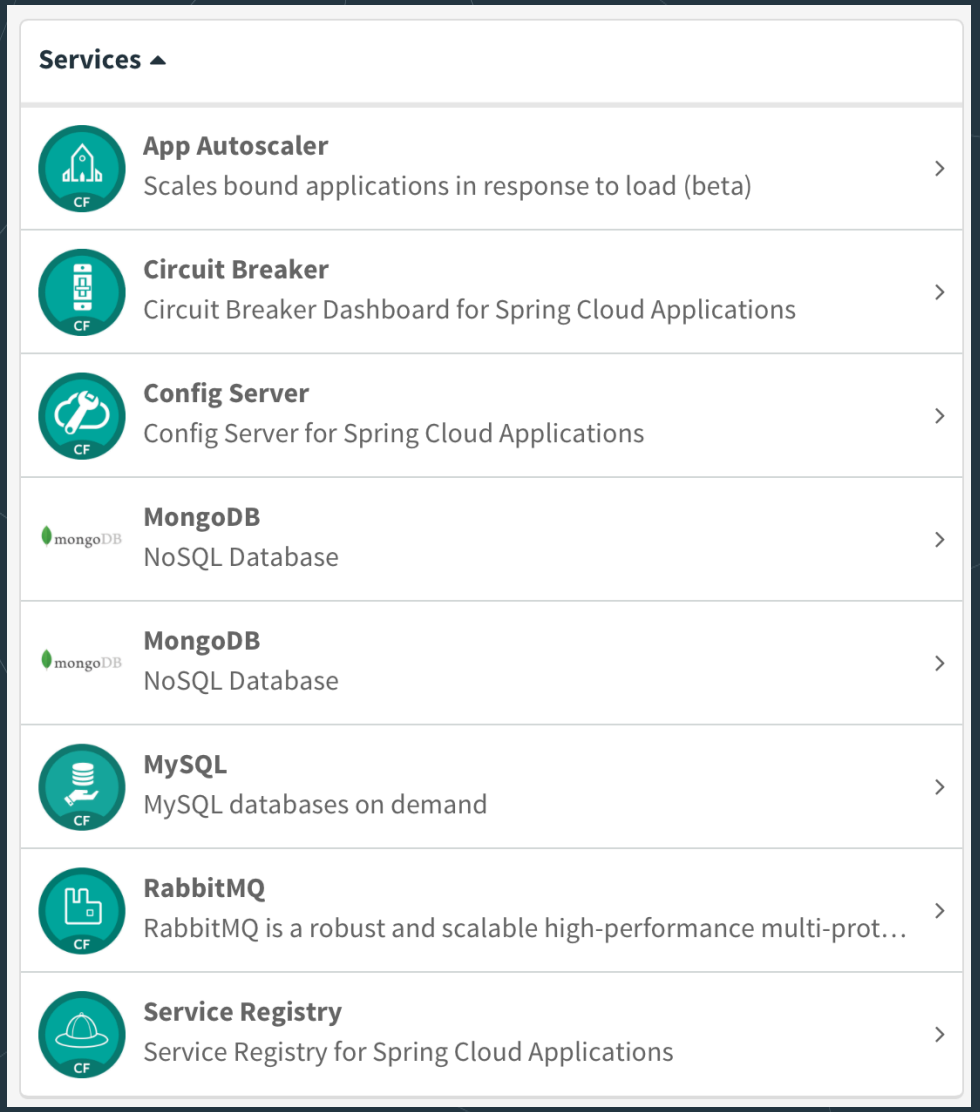
Pivotal

SECTION 4 CF MARKETPLACE & CF BIND

“Marketplace” of Managed Services

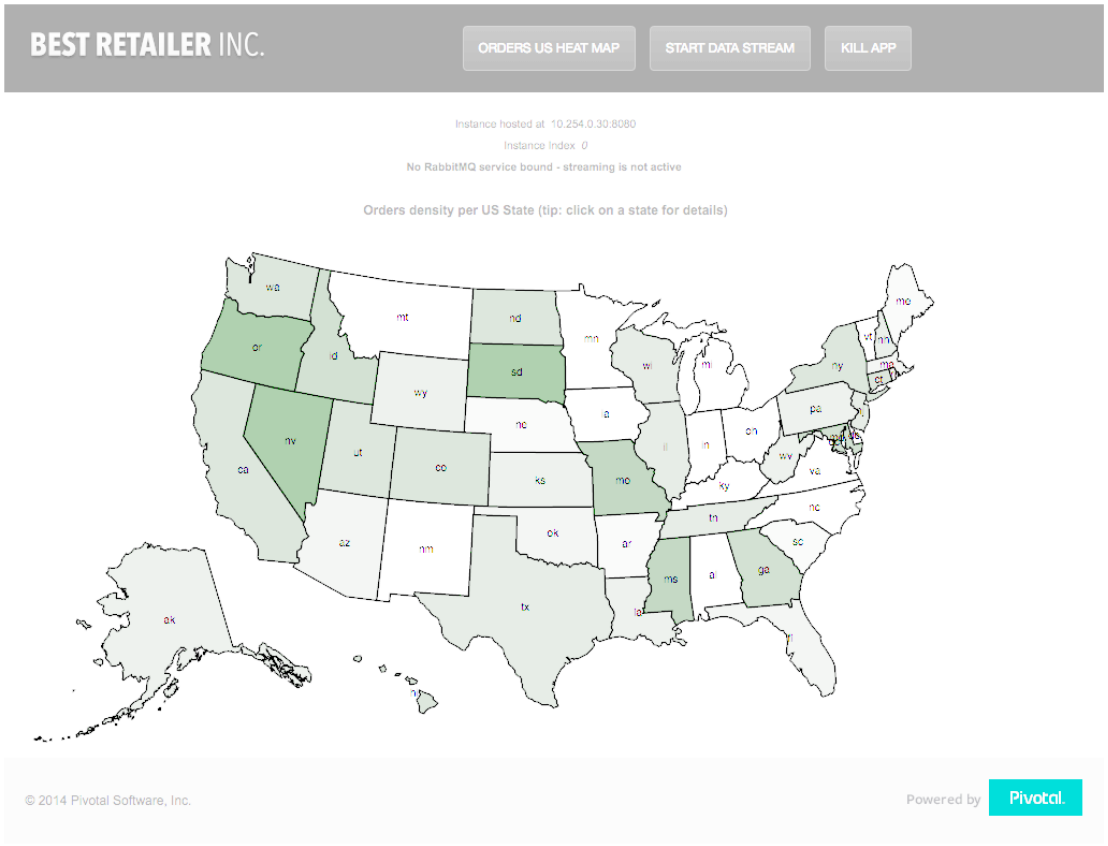
Managed Services are integrated with PCF by implementing a documented API.

A **self-service** catalog of databases, analytics and middleware technologies:

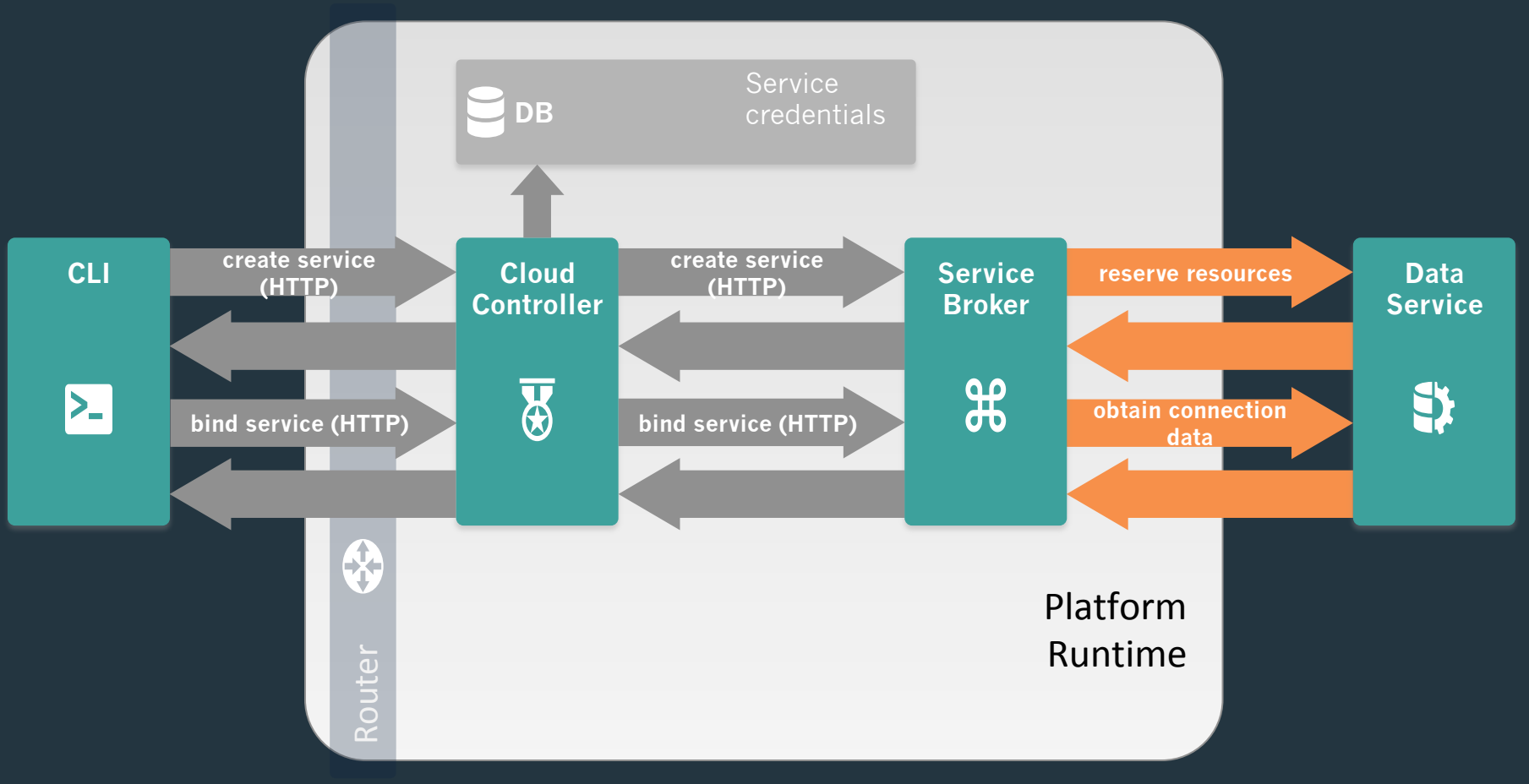


Pivotal

Push PCFDemo ….and Bind a RabbitMQ service



Creating and binding services



What about Services NOT in the marketplace?

- Hard-code Connection Info
- Environment Variables
- User-Provided Services:

```
MGunter-MB-Pro:demos mgunter$ cf cups my-secret-db -p "username, password, url"

username> mgunter

password> asdfasdfa

url> http://whereever
Creating user provided service my-secret-db in org Southeast / space mgunter-sp
OK
```

Pivotal

User-Provided Services

```
EXAMPLES:
   cf create-user-provided-service my-db-mine -p "username, password"
   cf create-user-provided-service my-db-mine -p /path/to/credentials.json
   cf create-user-provided-service my-drain-service -l syslog://example.com
   cf create-user-provided-service my-route-service -r https://example.com

   Linux/Mac:
      cf create-user-provided-service my-db-mine -p '{"username":"admin","password":"pa55woRD"}'

   Windows Command Line:
      cf create-user-provided-service my-db-mine -p "{\"username\":\"admin\",\"password\":\"pa55woRD\"}"
```

Pivotal

Binding Services to Apps

```
$ cf bind-service my-app mydb
Binding service mydb to my-app in org my-org / space test as me@example.com...
OK
TIP: Use 'cf push' to ensure your env variable changes take effect

$ cf restart my-app
```


Note: You must restart or in some cases re-push your application for changes to be applied to the VCAP_SERVICES environment variable and for the application to recognize these changes.

Pivotal

Session 4:

```
$ ls
pcfdemo.war
$ cf  push  [name]-demo –p pcfdemo.war
```

[browse to url created for app.]

```
$ cf  marketplace
$ cf  create-service p-rabbitmq standard [name]-RMQ
$ cf  bind-service [name]-demo [name]-RMQ

$ cf  create-user-provided-service  [name]-UPS -p "username, password, url”

$ cf  bind-service [name]-demo [name]-UPS
$ cf  env [name]-demo
```

Pivotal

SECTION 5 CF ENV

Env Vars

System-Provided:


```
{
 "VCAP_APPLICATION": {
  "application_id": "e6bc1d1e-4383-4311-972c-3dd33e278ddb",
  "application_name": "circuit-breaker-dashboard",
  "application_uris": [
   "hystrix-dashboard-meristematic-nytril.cfapps.io"
  ],
  "application_version": "aeab9221-4c79-4b7c-b514-346b7723ab91",
  "limits": {
   "disk": 1024,
   "fds": 16384,
   "mem": 384
  },
  "name": "circuit-breaker-dashboard",
  "space_id": "d20566e3-67c4-43e3-8c27-19f8951424be",
  "space_name": "experimental",
  "uris": [
   "hystrix-dashboard-meristematic-nytril.cfapps.io"
  ],
  "users": null,
  "version": "aeab9221-4c79-4b7c-b514-346b7723ab91"
 }
}
```


User-Provided:

```
MY_ENV_VAR: some-random-string
```

Pivotal

Other Environment Variables

ii. Application-Specific System Variables
- CF_INSTANCE_ADDR
- CF_INSTANCE_GUID
- CF_INSTANCE_INDEX
- CF_INSTANCE_IP
- CF_INSTANCE_PORT
- CF_INSTANCE_PORTS
- HOME
- MEMORY_LIMIT
- PORT
- PWD
- TMPDIR
- USER
- VCAP_APP_HOST
- VCAP_APP_PORT
- VCAP_APPLICATION
- VCAP_SERVICES

Pivotal

Session 5:
$ cf env [name]-spring-music

[Option - bind a new database service to this app and restart it]

$ cf env [name]-spring-music
$ cf set-env [env_name] [env_value]

$cf env [name]-spring-music

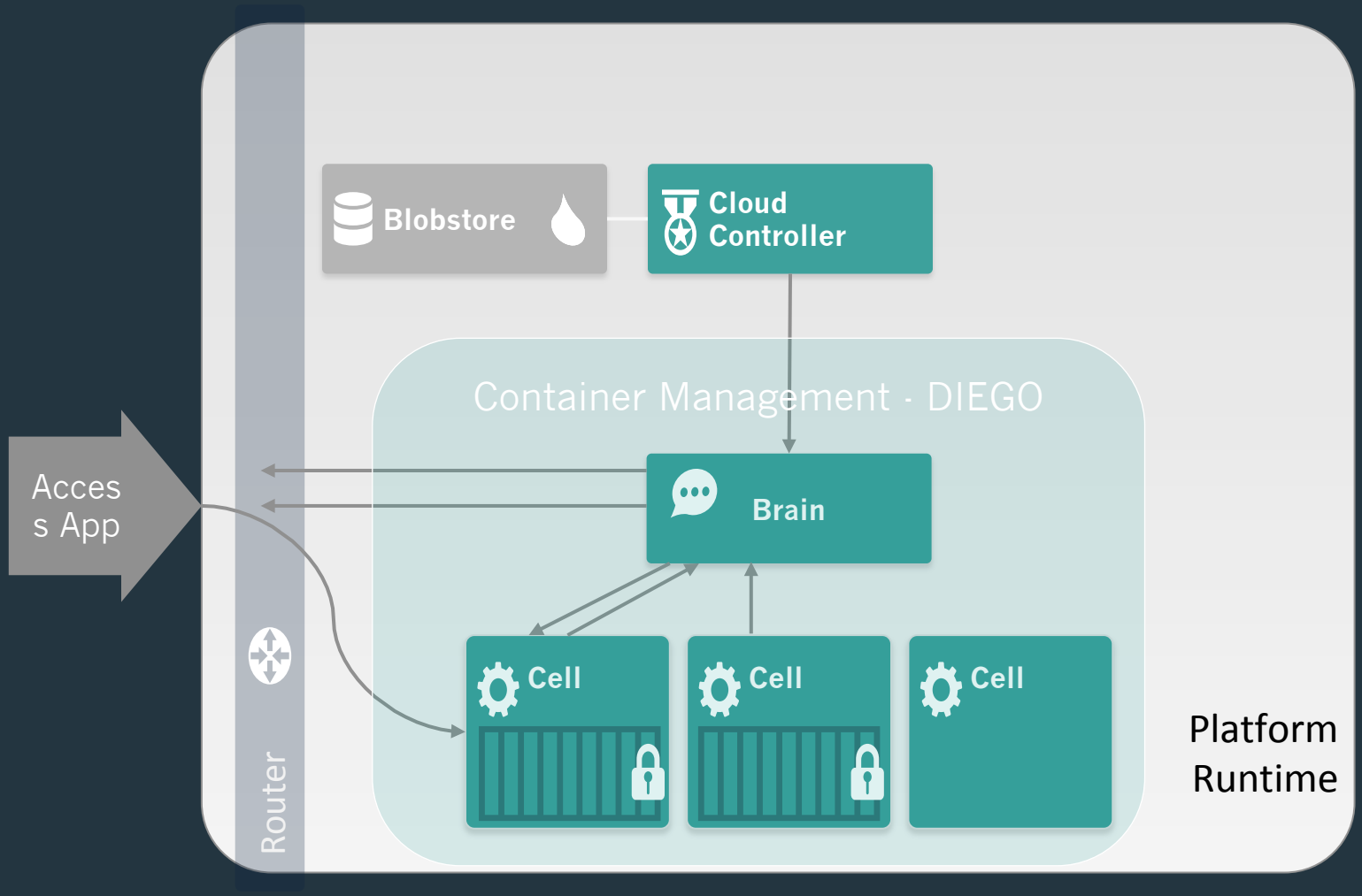
$cf restart [name]-spring-music

[view the (i) in the browser to confirm the database is being used]

Pivotal

SECTION 6 CF SCALE & QUOTA/ROLES

The App-Container Approach - Scaling



Organizations

- Top-most administrative unit
- Logical division division
- Each organization has its own users and assigned quota
- User permissions / Roles are specified per space within an organization
- Sub-divided into spaces

Spaces

- Logical sub-division within an organization
- Users authorized at an organization level can have different roles per space
- Services, Routes, and Applications are created / target per Space

- Can also have “space quota”

Session 6:

```
$ cf  scale [name]-demo -i 3
$ cf  app [name]-demo

$ cf  target (optional to verify your org and space info)

$ cf  org [org name]

$ cf  space [space name]
```

Pivotal

SECTION 7 HEALTH MONITORING & TROUBLESHOOTING

Viewing your app “vitals”

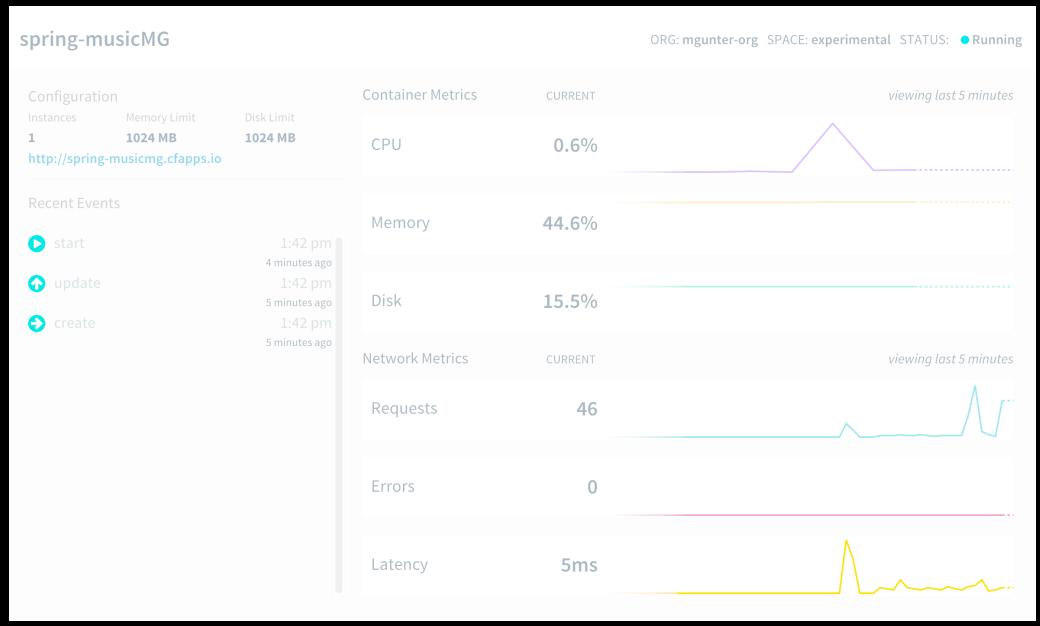
- cf events [app_name]
- cf app [app_name]
- cf logs [app_name] --recent

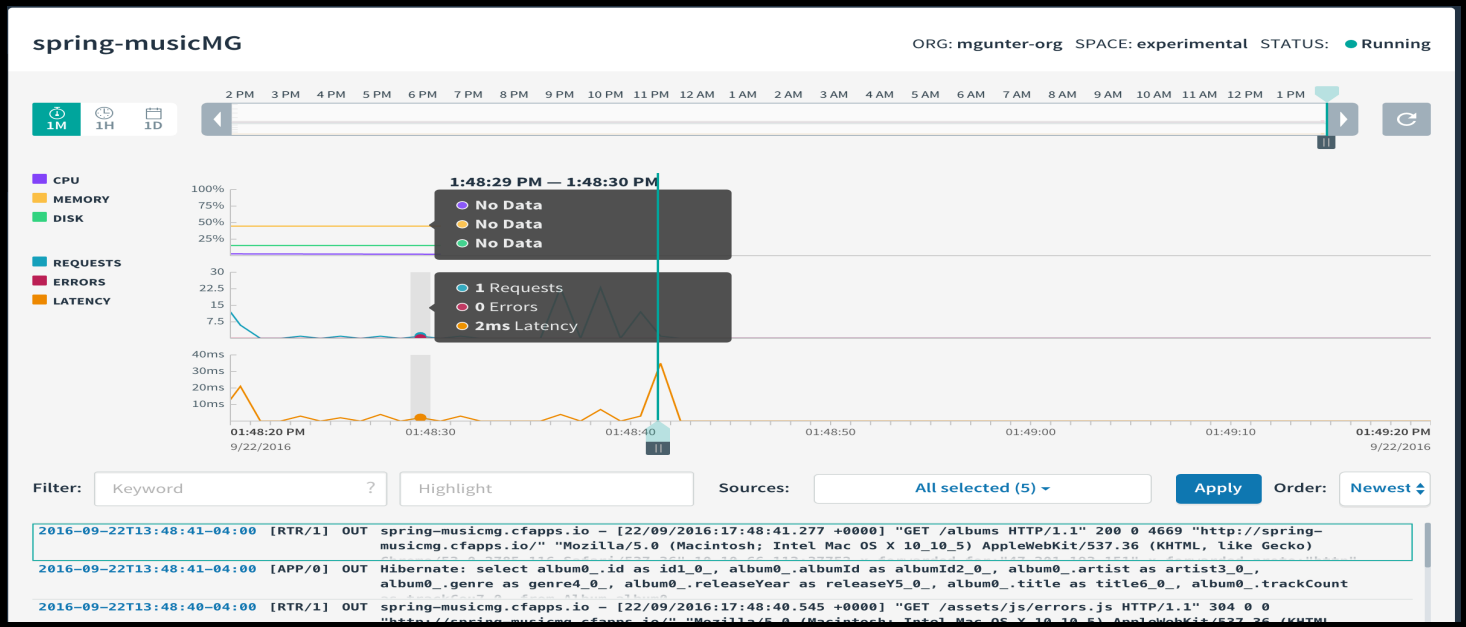
- PCF Metrics page

Pivotal

PCF Metrics….

Detailed performance info about your app.





Pivotal

Session 7:

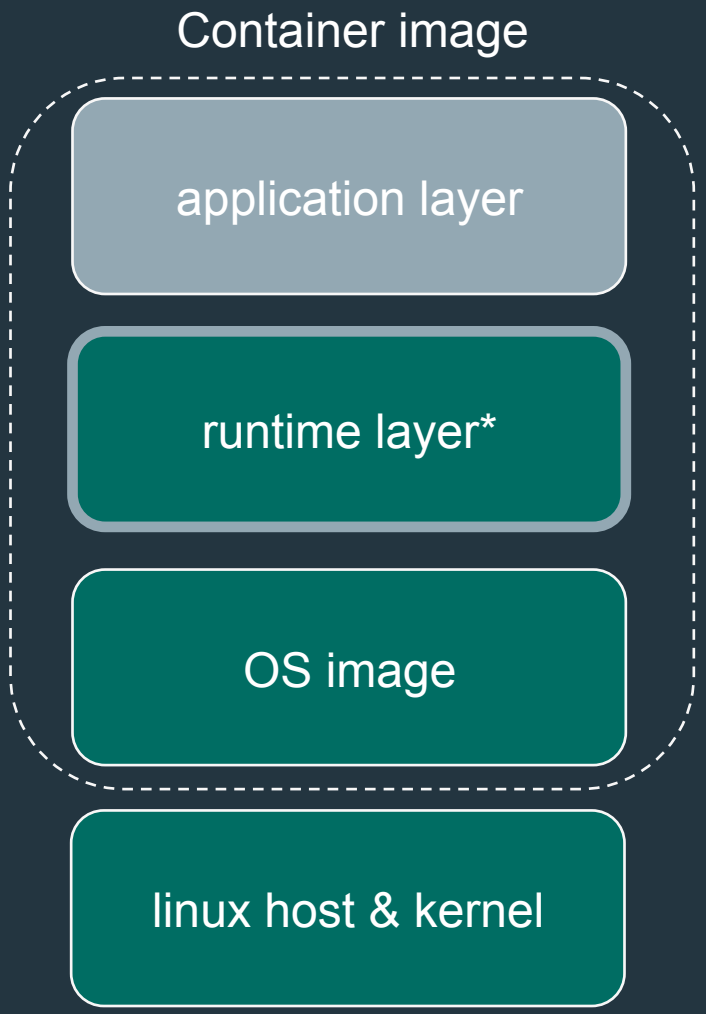
```
$ cf logs [name]-demo --recent
$ cf app [name]-demo
$ cf events [name]-demo
```

[browse to the PCF Metrics page via Apps Manager]
apps.paas.mia.ulti.io

Pivotal

SECTION 8 LANGUAGES & BUILDPACKS

Platform Runtime – Buildpacks



Dev Supported

PCF Supported

App container

Ops Supported

Pivotal CF supports
- **6 Language Buildpacks**
- Java, Node, Python, Ruby, Go, .Net Core(beta) , Static_file

Devs can focus on (& push) just the app code

PCF creates the container image for you

* Devs may bring a custom buildpack

Push your own Example App

Use one of the examples:

```
MGunter-MB-Pro:sample-apps mgunter$ tree -L
.
├── dotnetcore-app
├── go-sample-app
├── node-app
├── python-app
├── rails-sample-app
├── spring-app
└── staticfile-app
```

Make a change and re-push.

Pivotal

SECTION 9 WHERE TO GO FOR HELP?

Informational Resources:

UltiHome (search for “cloud engine”)

Ultimate Software SLACK channel: #48hrs-pcf

CF Docs: https://docs.cloudfoundry.org/

PCF Docs: https://docs.pivotal.io/pivotalcf

CF Basics Tutorial: http://basics-workshop.cloudfoundry.org/

Pivotal

Support for PCF Buildpacks, Spring, and Tomcat

runtime layer

Buildpacks support

- Java, Node, Python, Ruby, Go, .Net Core(beta), Static_file

Enterprise Middleware Support

- Apache Tomcat Support from #1 Committer

Enterprise Framework Support

- Spring is the #1 Enterprise Java Framework

Support.pivotal.io

Informational Resources:

UltiHome (search for “cloud engine”)

Ultimate Software SLACK channel: #48hrs-pcf

CF Docs: https://docs.cloudfoundry.org/

PCF Docs: https://docs.pivotal.io/pivotalcf

CF Basics Tutorial: http://basics-workshop.cloudfoundry.org/

Pivotal

Session 1:
Download CLI via browser from
https://apps.paas.mia.ulti.io/tools

[Install CLI]

Session 2:
$ cf login -a https://api.paas.mia.ulti.io
$ cf target
API endpoint: https://api.paas.mia.ulti.io (API version: 2.54.0)
User: training1
Org: 48hours
Space: training1
$ cf help *[optional]*

Session 3:
$ ls
spring-music.war
$ cf push [name]-spring-music -p spring-music.war
[browse to url created for app.]

$ cf map-route [name]-spring-music apps.mia.ulti.io --hostname [new-name]
[browse to url created for route.]

Session 4:
$ ls
pcfdemo.war
$ cf push [name]-demo -p pcfdemo.war
[browse to url created for app.]
$ cf marketplace
$ cf create-service p-rabbitmq standard [name]-RMQ
$ cf bind-service [name]-demo [name]-RMQ

$ cf create-user-provided-service [name]-UPS
-p "username, password, url"
$ cf bind-service [name]-demo [name]-UPS
$ cf env [name]-demo

Session 5:
$ cf env [name]-spring-music

[bind a new database service to this app and restart it]

$ cf env [name]-spring-music
$ cf set-env [name]-spring-music [env_name] [env_value]

$cf env [name]-spring-music

$cf restart [name]-spring-music

[view the (i) in the browser to confirm the database is being used]

Session 6:
$ cf scale [name]-demo -i 3
$ cf target *(optional to verify your org and space info)*

Session 7:
$ cf logs [name]-demo --recent
$ cf app [name]-demo
$ cf events [name]-demo
[browse to the PCF Metrics page via Apps Manager]