Pivotal™

Course developed by
Pivotal Academy

Pivotal Cloud Foundry Developer v1.7

SERVICES AND MANIFESTS

Agenda

1. Cloud Native Apps
2. Managed Services
3. User Provided Service Instances

The 12 Factors

Codebase	Dependencies	Config	Backing Services
Build, release, run	Processes	Port binding	Concurrency
Disposability	De/prod parity	Logs	Admin Processes

Config

Store config in the environment.

We should be storing configuration values in OS environment variables

Backing Services

Treat backing services as attached resources.

Agenda

1. Cloud Native Apps
2. Managed Services
3. User Provided Service Instances

```
DROBERTS-MBPRO:attendee-service droberts$ cf push attendee-service -p ./attendee-service-0.0.1-SNAPSHOT.jar -m 512M
Creating app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

Creating route attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io...
OK

Binding attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io to attendee-service...
OK

Uploading attendee-service...
Uploading app files from: /var/folders/84/ldbx2c5j01l_ycgg3d37g9yh0000gq/T/unzipped-app607471476
Uploading 826.3K, 109 files
Done uploading
```

We have already downloaded our ***attendee-service-0.0.1-SNAPSHOT.jar*** file, we then push it using the ***$ cf push attendee-service-0.0.1-SNAPSHOT.jar -m 512M - -random-route*** command

```
Successfully created container
Downloading app package...
Downloaded app package (26.3M)
Staging...
-----> Java Buildpack Version: v3.6 (offline) | https://github.com/cloudfoundry/java-buildpack.git#5194155
-----> Downloading Open Jdk JRE 1.8.0_71 from https://download.run.pivotal.io/openjdk/trusty/x86_64/openjdk-1.8.0_71
 cache)
       Expanding Open Jdk JRE to .java-buildpack/open_jdk_jre (1.0s)
-----> Downloading Open JDK Like Memory Calculator 2.0.1_RELEASE from https://download.run.pivotal.io/memory-calcula
/memory-calculator-2.0.1_RELEASE.tar.gz (found in cache)
       Memory Settings: -Xms382293K -Xss995K -Xmx382293K -XX:MaxMetaspaceSize=64M -XX:MetaspaceSize=64M
-----> Downloading Spring Auto Reconfiguration 1.10.0_RELEASE from https://download.run.pivotal.io/auto-reconfigurat
uration-1.10.0_RELEASE.jar (found in cache)
Exit status 0
Staging complete
Uploading droplet, build artifacts cache...
Uploading build artifacts cache...
Uploading droplet...
Uploaded build artifacts cache (108B)
Uploaded droplet (71.2M)
Uploading complete

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 crashed
FAILED
Start unsuccessful

TIP: use 'cf logs attendee-service --recent' for more information
DROBERTS-MBPRO:attendee-service droberts$
```

We see that the service failed to start

```
DROBERTS-MBPRO:attendee-service droberts$ cf logs attendee-service --recent
Connected, dumping recent logs for app [redacted] in org [redacted] / space [redacted] as [redacted]...
```

```
operties$DataSourceBeanCreationException: Cannot determine embedded database driver class for
dded database please put a supported one on the classpath. If you have database settings to be
ay need to active it (the profiles "cloud" are currently active).
17:36:37.14-0500 [APP/0]      ERR    at org.springframework.beans.factory.support.SimpleInstan
ntiationStrategy.java:189)
17:36:37.14-0500 [APP/0]      ERR    at org.springframework.beans.factory.support.ConstructorR
onstructorResolver.java:588)
17:36:37.14-0500 [APP/0]      ERR    ... 216 more
17:36:37.14-0500 [APP/0]      ERR Caused by: org.springframework.boot.autoconfigure.jdbc.DataS
      Cannot determine embedded database driver class for database type NONE. If you want
rted one on the classpath. If you have database settings to be loaded from a particular profi
es "cloud" are currently active).
17:36:37.14-0500 [APP/0]      ERR    at org.springframework.boot.autoconfigure.jdbc.DataSource
eProperties.java:180)
17:36:37.14-0500 [APP/0]      ERR    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Me
17:36:37.14-0500 [APP/0]      ERR    at sun.reflect.NativeMethodAccessorImp[redacted]ke(NativeMeth
17:36:37.14-0500 [APP/0]      ERR    at sun.reflect.DelegatingMethodAccess[redacted]invoke(Delega

17:36:37.14-0500 [APP/0]      ERR    at java.lang.reflect.Method.invoke(Met[redacted]va:497)
17:36:37.14-0500 [APP/0]      ERR    at org.springframework.beans.factory.[redacted]SimpleInstan
ntiationStrategy.java:162)
17:36:37.14-0500 [APP/0]      ERR    ... 217 more
17:36:37.17-0500 [APP/0]      OUT Exit status 0
17:36:37.18-0500 [CELL/0]     OUT Exit status 0
17:36:37.19-0500 [API/0]      OUT App instance exited with [redacted]
```

This tells us that the application needs a DB to work properly. This means that we need to provision a DB instance and bind it to the attendee-service before restarting it again.

```
DROBERTS-MBPRO:attendee-service droberts$ cf m
Getting services from marketplace in org dave / space dev as droberts@pivotal.io...
OK

service          plans                     description
app-autoscaler   bronze, gold              Scales bound applications in response to load
p-mysql          100mb-dev                 MySQL service for application development and testing
p-redis          shared-vm, dedicated-vm   Redis service to provide a key-value store

TIP:  Use 'cf marketplace -s SERVICE' to view descriptions of individual plans of a given service.
DROBERTS-MBPRO:attendee-service droberts$ cf create-service p-mysql 100mb-dev attendee-mysql
Creating service instance attendee-mysql in org dave / space dev as droberts@pivotal.io...
OK
DROBERTS-MBPRO:attendee-service droberts$
```

We can use the ***$ cf m*** command to see what services are available to be loaded from the Marketplace. Then we use the ***$ cf create-service p-mysql 100mb-dev attendee-mysql*** command to provision a MySQL service instance called ***attendee-mysql*** as our DB service to use with the attendee-service.

```
DROBERTS-MBPRO:attendee-service droberts$ cf create-service p-mysql 100mb-dev attendee-mysql
Creating service instance attendee-mysql in org dave / space dev as droberts@pivotal.io...
OK
DROBERTS-MBPRO:attendee-service droberts$ cf bind-service attendee-service attendee-mysql
Binding service attendee-mysql to app attendee-service in org dave / space dev as droberts@pivotal.io...
OK
TIP: Use 'cf restage attendee-service' to ensure your env variable changes take effect
DROBERTS-MBPRO:attendee-service droberts$
```

Then we bind the ***attendee-mysql*** DB service to our ***attendee-service*** using the ***$ cf bind-service attendee-service attendee-mysql*** command above.

```
DROBERTS-MBPRO:attendee-service droberts$ cf m
Getting services from marketplace in org dave / space dev as droberts@pivotal.io...
OK

service          plans                     description
app-autoscaler   bronze, gold              Scales bound applications in response to load
p-mysql          100mb-dev                 MySQL service for application development and testing
p-redis          shared-vm, dedicated-vm   Redis service to provide a key-value store

TIP:  Use 'cf marketplace -s SERVICE' to view descriptions of individual plans of a given service.
DROBERTS-MBPRO:attendee-service droberts$ cf create-service p-mysql 100mb-dev attendee-mysql
Creating service instance attendee-mysql in org dave / space dev as droberts@pivotal.io...
OK
DROBERTS-MBPRO:attendee-service droberts$ cf bind-service attendee-service attendee-mysql
Binding service attendee-mysql to app attendee-service in org dave / space dev as droberts@pivotal.io...
OK
TIP: Use 'cf restage attendee-service' to ensure your env variable changes take effect
DROBERTS-MBPRO:attendee-service droberts$ cf restart attendee-service
Stopping app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

Starting app attendee-service in org dave / space dev as droberts@pivotal.io...

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
```

We then use the ***$ cf restart attendee-service*** command to restart the attendee-service to make sure all the environment variables are correctly used and that our app instances parse them for connecting to the DB.

```
Starting app attendee-service in org dave / space dev as droberts@pivotal.io...

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App attendee-service was started using this command `CALCULATED_MEMORY=$($PWD/.java-buildpack/open_jdk_jre/bin/java-
calculator-2.0.1_RELEASE -memorySizes=metaspace:64m.. -memoryWeights=heap:75,metaspace:10,native:10,stack:5 -memoryI
,metaspace:100% -totMemory=$MEMORY_LIMIT) && JAVA_OPTS="-Djava.io.tmpdir=$TMPDIR -XX:OnOutOfMemoryError=$PWD/.java-b
_jre/bin/killjava.sh $CALCULATED_MEMORY" && SERVER_PORT=$PORT eval exec $PWD/.java-buildpack/open_jdk_jre/bin/java $
D/.:$PWD/.java-buildpack/spring_auto_reconfiguration/spring_auto_reconfiguration-1.10.0_RELEASE.jar org.springframew
arLauncher`

Showing health and status for app attendee-service in org dave / space dev as droberts@pivotal...
OK

requested state: started
instances: 1/1
usage: 512M x 1 instances
urls: attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io
last uploaded: Wed May 25 22:35:34 UTC 2016
stack: unknown
buildpack: java-buildpack=v3.6-offline-https://github.com/cloudfoundry/java-buildpac                    en-jdk
1 open-jdk-like-memory-calculator=2.0.1_RELEASE spring-auto-reconfiguration=1.10.0

     state     since                    cpu    memory      disk      detai
```

```
DROBERTS-MBPRO:attendee-service droberts$ cf app attendee-service
Showing health and status for app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

requested state: started
instances: 1/1
usage: 512M x 1 instances
urls: attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io
last uploaded: Wed May 25 22:35:34 UTC 2016
stack: cflinuxfs2
buildpack: java-buildpack=v3.6-offline-https://github.com/cloudfoundry/java-buildpack.git#5194155 java-main open-jdk
1 open-jdk-like-memory-calculator=2.0.1_RELEASE spring-auto-reconfiguration=1.10.0_RELEASE

     state     since                    cpu    memory           disk           details
#0   running   2016-05-25 05:39:08 PM   0.0%   392.6M of 512M   152.4M of 1G
DROBERTS-MBPRO:attendee-service droberts$
```

We can use the ***$ cf app attendee-service*** command to see the health of our attendee-service, they are in a healthy state.

```
attendee-service-monochr ×
attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io

{
  "_links": {
    "attendees": {
      "href": "http://attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io/attendees{?page,size,sort}",
      "templated": true
    },
    "profile": {
      "href": "http://attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io/profile"
    }
  }
}
```

Our attendee-service REST API is up and running.

```
DROBERTS-MBPRO:attendee-service droberts$ cf env attendee-service
Getting env variables for app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

System-Provided:
{
 "VCAP_SERVICES": {
  "p-mysql": [
   {
    "credentials": {
     "hostname": "10.65.188.80",
     "jdbcUrl": "jdbc:mysql://10.65.188.80:3306/cf_9b4bba91_c2cc_4b82_a9ce_61506b60dce9?user=4otnLyRKEv0bz8r3\u0026p
xqLUjcn",
     "name": "cf_9b4bba91_c2cc_4b82_a9ce_61506b60dce9",
     "password": "lFf1XBQGYxqLUjcn",
     "port": 3306,
     "uri": "mysql://4otnLyRKEv0bz8r3:lFf1XBQGYxqLUjcn@10.65.188.80:3306/cf_9b4bba91_c2cc_4b82_a9ce_61506b60dce9?rec
     "username": "4otnLyRKEv0bz8r3"
    },
    "label": "p-mysql",
    "name": "attendee-mysql",
    "plan": "100mb-dev",
    "provider": null,
    "syslog_drain_url": null,
    "tags": [
     "mysql",
     "relational"
    ]
   }
  ]
 }
```

When you provision services for your applications, the service environment variables get put in your services **VCAP_SERVICES** environment variables as seen from using the ***$ cf env attendee-service*** command as above. We can see the MySQL service instance URI that our attendee-service will be connecting to, as an attached resource.

```
}

{
 "VCAP_APPLICATION": {
  "application_id": "b7418b24-b71e-4f4f-8ff5-8e1d6ef6cc04",
  "application_name": "attendee-service",
  "application_uris": [
   "attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io"
  ],
  "application_version": "cf84c2e4-d752-435b-a9e9-3e64f7d0c804",
  "limits": {
   "disk": 1024,
   "fds": 16384,
   "mem": 512
  },
  "name": "attendee-service",
  "space_id": "2cf2d8ab-6ad2-4ad3-9009-17128854ad7b",
  "space_name": "dev",
  "uris": [
   "attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io"
  ],
  "users": null,
  "version": "cf84c2e4-d752-435b-a9e9-3e64f7d0c804"
 }
}

No user-defined env variables have been set

No running env variables have been set

No staging env variables have been set
```

```
DROBERTS-MBPRO:attendee-service droberts$ cf create-user-provided-service attendee-service -p uri

uri>
```

We can now hook up the REST API **attendee-service** to our UI **articulate service** as **a user provided service** running within our CF environment. We simply create a user provided service called attendee-service and provide its URI using the ***$ cf create-user-provided-service attendee-service -p uri*** command.

```
DROBERTS-MBPRO:attendee-service droberts$ cf apps
Getting apps in org dave / space dev as droberts@pivotal.io...
OK

name               requested state   instances   memory   disk   urls
articulate         started           1/1         512M     1G     articulate-turbosupercharged-spinneret.cfapps.haas-
attendee-service   started           1/1         512M     1G     attendee-service-monochromatic-guarantee.cfapps.haa
io
DROBERTS-MBPRO:attendee-service droberts$
```

We can use the ***$ cf apps*** command to see the URI's of our running applications

```
DROBERTS-MBPRO:attendee-service droberts$ cf create-user-provided-service attendee-service -p uri

uri> http://attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io
Creating user provided service attendee-service in org dave / space dev as droberts@pivotal.io...
OK
DROBERTS-MBPRO:attendee-service droberts$
```

We now have our user provided service created. Next, we need to bind the UI articulate service application to our new user provided service of the attendee-service.

```
DROBERTS-MBPRO:attendee-service droberts$ cf bind-service articulate attendee-service
Binding service attendee-service to app articulate in org dave / space dev as droberts@pivotal.io...
OK
TIP: Use 'cf restage articulate' to ensure your env variable changes take effect
DROBERTS-MBPRO:attendee-service droberts$
```

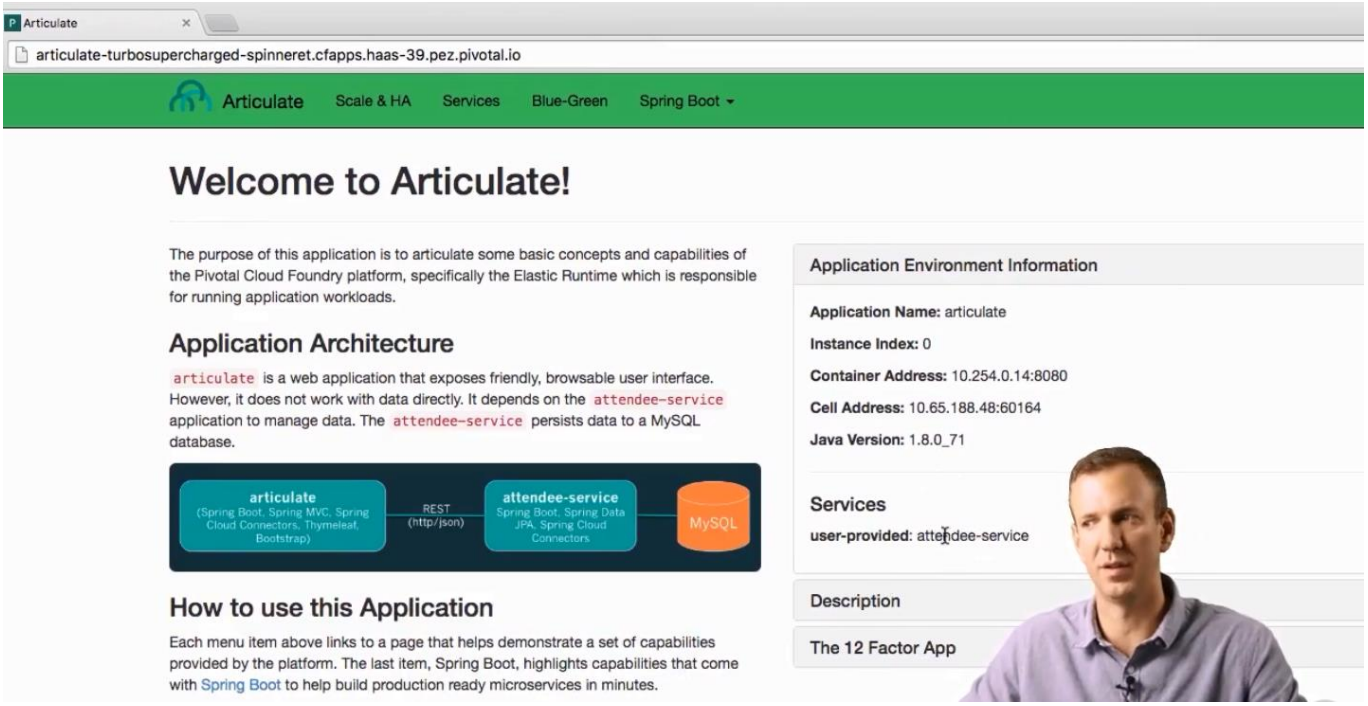
We bind the UI app to the user provided service for the attendee-service using the ***$ cf bind-service articulate attendee-service*** command.

```
Creating user provided service attendee-service in org dave / space dev as droberts@pivotal.io...
OK
DROBERTS-MBPRO:attendee-service droberts$ cf bind-service articulate attendee-service
Binding service attendee-service to app articulate in org dave / space dev as droberts@pivotal.io...
OK
TIP: Use 'cf restage articulate' to ensure your env variable changes take effect
DROBERTS-MBPRO:attendee-service droberts$ cf restart articulate
Stopping app articulate in org dave / space dev as droberts@pivotal.io...
OK

Starting app articulate in org dave / space dev as droberts@pivotal.io...

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App articulate was started using this command `CALCULATED_MEMORY=$($PWD/.java-buildpack/open  bin/java-buildp
ator-2.0.1_RELEASE -memorySizes=metaspace:64m.. -memoryWeights=heap:75,metaspace:10,native:1  5 -memoryInitial
pace:100% -totMemory=$MEMORY_LIMIT) && JAVA_OPTS="-Djava.io.tmpdir=$TMPDIR -XX:OnOutOfMemoryE  WD/.java-buildpa
in/killjava.sh $CALCULATED_MEMORY" && SERVER_PORT=$PORT eval exec $PWD/.java-buildpack/o  ava $JAVA_O
WD/.java-buildpack/spring_auto_reconfiguration/spring_auto_reconfiguration-1.10.0_RELE  ework.bo
cher`

Showing health and status for app articulate in org dave / space dev as droberts
```

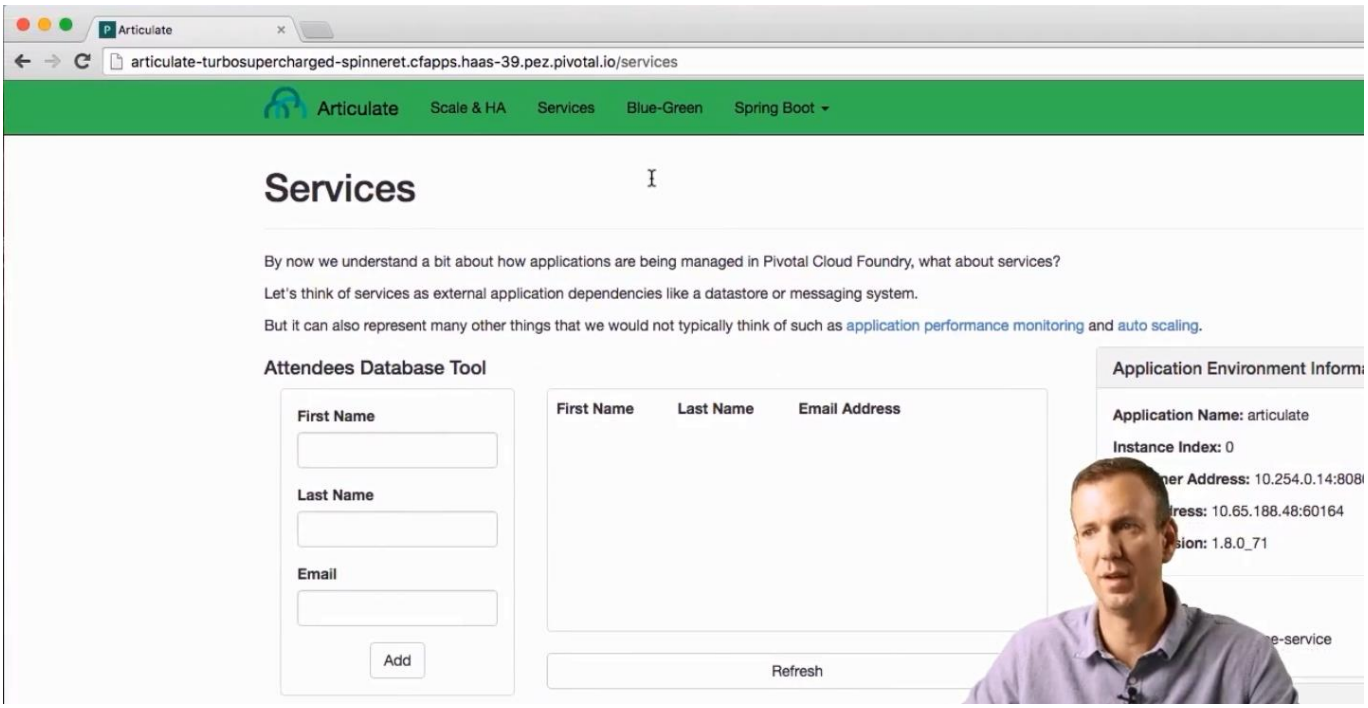
We then restart the articulate UI app

```
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App articulate was started using this command `CALCULATED_MEMORY=$($PWD/.java-buildpack/open_jdk_jre/bin/java-buildp
ator-2.0.1_RELEASE -memorySizes=metaspace:64m.. -memoryWeights=heap:75,metaspace:10,native:10,stack:5 -memoryInitial
pace:100% -totMemory=$MEMORY_LIMIT) && JAVA_OPTS="-Djava.io.tmpdir=$TMPDIR -XX:OnOutOfMemoryError=$PWD/.java-buildpa
in/killjava.sh $CALCULATED_MEMORY" && SERVER_PORT=$PORT eval exec $PWD/.java-buildpack/open_jdk_jre/bin/java $JAVA_O
WD/.java-buildpack/spring_auto_reconfiguration/spring_auto_reconfiguration-1.10.0_RELEASE.jar org.springframework.bo
cher`

Showing health and status for app articulate in org dave / space dev as droberts@pivotal.io...
OK

requested state: started
instances: 1/1
usage: 512M x 1 instances
urls: articulate-turbosupercharged-spinneret.cfapps.haas-39.pez.pivotal.io
last uploaded: Wed May 25 16:04:12 UTC 2016
stack: unknown
buildpack: java-buildpack=v3.6-offline-https://github.com/cloudfoundry/java-buildpack.gi  in open-jdk
1 open-jdk-like-memory-calculator=2.0.1_RELEASE spring-auto-reconfiguration=1.10.0_REL

     state     since                    cpu    memory           disk           det
#0   running   2016-05-25 05:43:56 PM   0.1%   366.3M of 512M   154.2M of 1G
DROBERTS-MBPRO:attendee-service droberts$
```

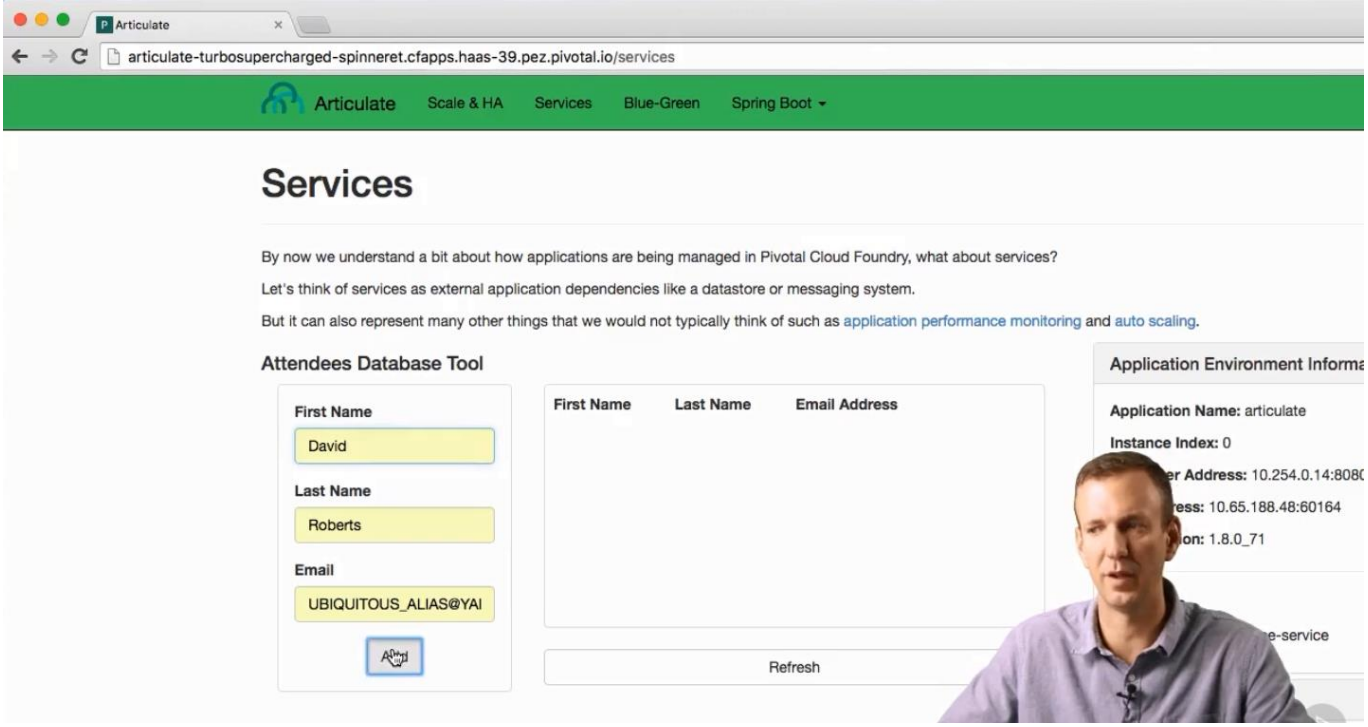
```
DROBERTS-MBPRO:attendee-service droberts$ cf apps
Getting apps in org dave / space dev as droberts@pivotal.io...
OK

name               requested state   instances   memory   disk   urls
articulate         started           1/1         512M     1G     articulate-turbosupercharged-spinneret.cfapps.haas-
attendee-service   started           1/1         512M     1G     attendee-service-monochromatic-guarantee.cfapps.haa
io
DROBERTS-MBPRO:attendee-service droberts$
```

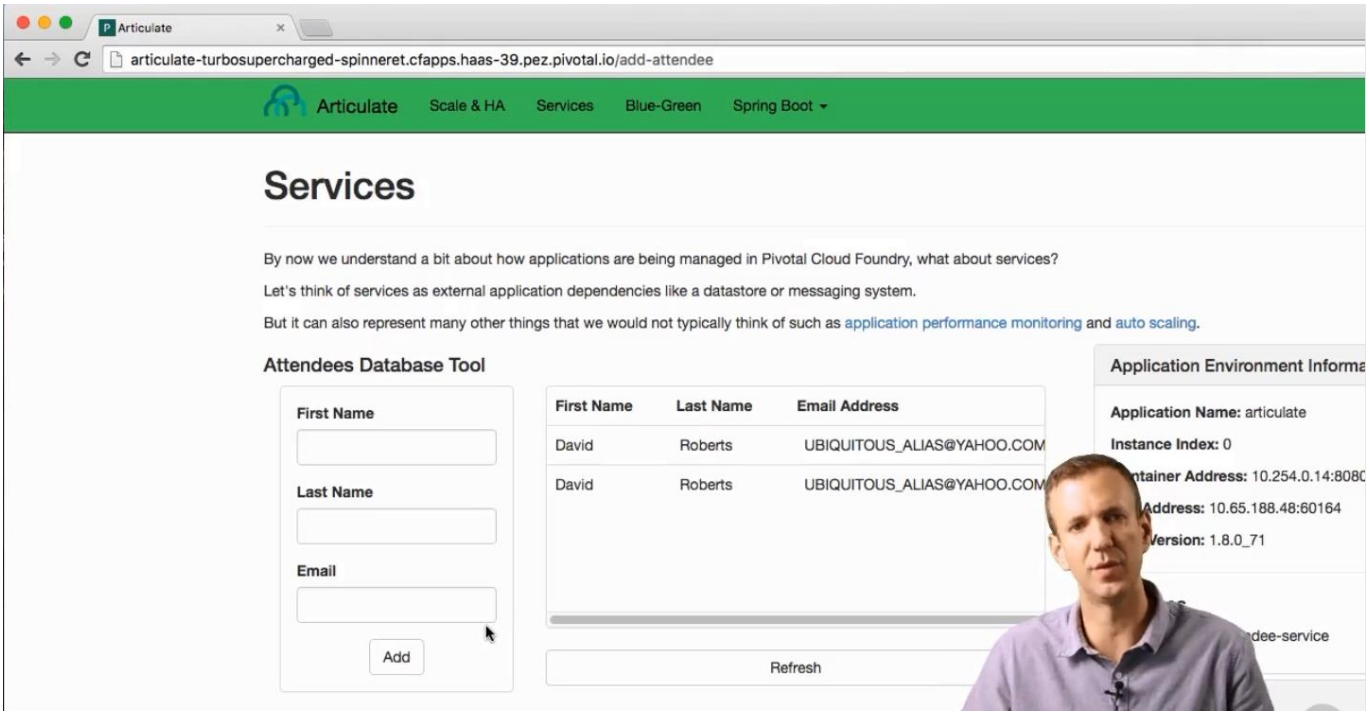


We can now see that we have a user provided service called attendee-service being used by the UI articulate app.



On our services page in the articulate app, we can now populate the data for different attendees and have the data persisted in our user provided service's MySQL DB.





This is how we can wire different services together like MySQL, Redis, Oracle cluster, ERP systems, etc.

Agenda

1. Cloud Native Apps
2. Managed Services
3. User Provided Service Instances

Marketplace

Managed services advertise a catalog of plans from which service instances can be provisioned.

```
mysql 100mb
redis dedicated
```

Service

Manages service instance lifecycle.

```
creating
binding
unbinding
deleting
```

Service Instance

A reserved resource such as a database on a shared or dedicated system.

VCAP_SERVICES


```
"VCAP_SERVICES": {
 "p-mysql": [
  {
   "credentials": {
    "hostname": "10.68.104.83",
    "jdbcUrl": "jdbc:mysql://10.68.104.83:3306/cf_58ad484c_f6c3_4f5f_8acc_e623bc8e302b?user=jlipqkmQqyaBlefS
\u0026password=gm5o2PWsmyKqRn8y",
    "name": "cf_58ad484c_f6c3_4f5f_8acc_e623bc8e302b",
    "password": "gm5o2PWsmyKqRn8y",
    "port": 3306,
    "uri": "mysql://jlipqkmQqyaBlefS:gm5o2PWsmyKqRn8y@10.68.104.83:3306/cf_58ad484c_f6c3_4f5f_8acc_e623bc8e302b?reconnect=true",
    "username": "jlipqkmQqyaBlefS"
   },
   "label": "p-mysql",
   "name": "attendee-mysql",
   "plan": "100mb-dev",
   "tags": [
    "mysql",
    "relational"
   ]
  }
 ]
...
```


The environment variables are stored in the VCAP_SERVICES variable

Agenda

1. Cloud Native Apps
2. Managed Services
3. User Provided Service Instances

User Provided Service Instance

Enable developers to use services that are not available in the Marketplace with their applications running on Pivotal Cloud Foundry.

Oracle

DB2

User Provided Service Instance

Also used for application to application binding.

VCAP_SERVICES


```
"VCAP_SERVICES": {
...
 "user-provided": [
  {
   "credentials": {
    "uri": "oracle://root:secret@dbserver.example.com:1521/mydatabase"
   },
   "label": "user-provided",
   "name": "oracle-db",
   "syslog_drain_url": "",
   "tags": []
  }
 ]
```


Services

Recap

```
DROBERTS-MBPRO:attendee-service droberts$ cf | grep manifest
   create-app-manifest                    Create an app manifest for an app that has been pushed successfully
DROBERTS-MBPRO:attendee-service droberts$
```

Let us now see the concept of manifests. Manifests in **manifest.yml** file lets you push your applications without using the CLI but instead storing the needed values in a file attached to your application. The manifest is not uploaded with your application bits, instead it is used and parsed by the CF CLI to send multiple REST requests/commands to the cloud controller to upload your application and start it.

```
DROBERTS-MBPRO:attendee-service droberts$ cf create-app-manifest attendee-service -p ./manifest.yml
Creating an app manifest from current settings of app attendee-service ...

OK
Manifest file created successfully at ./manifest.yml

DROBERTS-MBPRO:attendee-service droberts$
```

We use the ***$ cf create-app-manifest attendee-service -p ./manifest.yml*** command to create a new **manifest.yml** file for our attendee-service REST API in the directory specified as above.

```
DROBERTS-MBPRO:attendee-service droberts$ cat manifest.yml
applications:
- name: attendee-service
  instances: 1
  memory: 512M
  disk_quota: 1024M
  host: attendee-service-monochromatic-guarantee
  domain: cfapps.haas-39.pez.pivotal.io
  services:
  - attendee-mysql
  stack: cflinuxfs2
DROBERTS-MBPRO:attendee-service droberts$
```

Use the ***$ cat manifest.yml*** command to see all the parameters needed to create and start this attendee-service application. Stacks refer to the backend to run this application on within PCF like Linux or windows.

```
DROBERTS-MBPRO:attendee-service droberts$ nano manifest.yml
```

```
  GNU nano 2.0.6                          File: manifest.yml

applications:
- name: attendee-service
  instances: 1
  memory: 512M
  disk_quota: 1024M
  host: attendee-service-monochromatic-guarantee
  domain: cfapps.haas-39.pez.pivotal.io
  services:
  - attendee-mysql
  stack: cflinuxfs2
```

We can also modify our manifest.yml file and include more details as below

```
DROBERTS-MBPRO:attendee-service droberts$ ls
attendee-service-0.0.1-SNAPSHOT.jar     manifest.yml
DROBERTS-MBPRO:attendee-service droberts$
```

```
  GNU nano 2.0.6                          File: manifest.yml

applications:
- name: attendee-service
  instances: 2
  memory: 512M
  disk_quota: 1024M
  host: attendee-service-monochromatic-guarantee
  domain: cfapps.haas-39.pez.pivotal.io
  path: ./attendee-service-0.0.1-SNAPSHOT.jar
  services:
  - attendee-mysql
  stack: cflinuxfs2
```

We can include the local path to the application JAR file in the **path** variable inside the manifest.yml file as above

```
DROBERTS-MBPRO:attendee-service droberts$ cf push
Using manifest file /Users/droberts/pcf-developer-workshop/attendee-service/manifest.yml

Using stack cflinuxfs2...
OK
Updating app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

Using route attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io
Uploading attendee-service...
Uploading app files from: /var/folders/84/ldbx2c5j01l_ycgg3d37g9yh0000gq/T/unzipped-app687378285
Uploading 826.3K, 109 files
Done uploading
```

We then do a ***$ cf push*** command to deploy the app, we can see that the first line retrieves and reads the attached manifest.yml file for the configuration details the app needs to start up.

```
Downloading java_buildpack_offline...
Downloading staticfile_buildpack...
Downloaded php_buildpack
Downloaded java_buildpack_offline
Downloaded python_buildpack
Downloaded nodejs_buildpack
Downloaded staticfile_buildpack
Downloaded ruby_buildpack
Downloaded go_buildpack
Downloaded binary_buildpack
Creating container
Successfully created container
Downloading app package...
Downloaded app package (26.3M)
Downloading build artifacts cache...
Downloaded build artifacts cache (108B)
Staging...
-----> Java Buildpack Version: v3.6 (offline) | https://github.com/cloudfoundry/java-buildpack.git#5194155
-----> Downloading Open Jdk JRE 1.8.0_71 from https://download.run.pivotal.io/openjdk/trus[illegible]64/openjdk-1.8.0_71
 cache)
       Expanding Open Jdk JRE to .java-buildpack/open_jdk_jre (1.0s)
-----> Downloading Open JDK Like Memory Calculator 2.0.1_RELEASE from https://download.ru[illegible]io/memory-calcula
/memory-calculator-2.0.1_RELEASE.tar.gz (found in cache)
       Memory Settings: -Xmx382293K -XX:MaxMetaspaceSize=64M -Xss995K -Xms382293K -XX:Meta[illegible]ze=64M
-----> Downloading Spring Auto Reconfiguration 1.10.0_RELEASE from https://download.run.pi[illegible]/auto-reconfigurat
uration-1.10.0_RELEASE.jar (found in cache)
Exit status 0
Staging complete
Uploading droplet, build artifacts cache...
Uploading build artifacts cache...
Uploading droplet...
```

```
0 of 2 instances running, 2 starting
2 of 2 instances running

App started


OK

App attendee-service was started using this command `CALCULATED_MEMORY=$($PWD/.java-buildpack/open_jdk_jre/bin/java-b
calculator-2.0.1_RELEASE -memorySizes=metaspace:64m.. -memoryWeights=heap:75,metaspace:10,native:10,stack:5 -memoryIn
,metaspace:100% -totMemory=$MEMORY_LIMIT) && JAVA_OPTS="-Djava.io.tmpdir=$TMPDIR -XX:OnOutOfMemoryError=$PWD/.java-bu
_jre/bin/killjava.sh $CALCULATED_MEMORY" && SERVER_PORT=$PORT eval exec $PWD/.java-buildpack/open_jdk_jre/bin/java $J
D/.:$PWD/.java-buildpack/spring_auto_reconfiguration/spring_auto_reconfiguration-1.10.0_RELEASE.jar org.springframewo
arLauncher`

Showing health and status for app attendee-service in org dave / space dev as droberts@pivotal.io...
OK

requested state: started
instances: 2/2
usage: 512M x 2 instances
urls: attendee-service-monochromatic-guarantee.cfapps.haas-39.pez.pivotal.io
last uploaded: Fri May 27 21:36:43 UTC 2016
stack: unknown
buildpack: java-buildpack=v3.6-offline-https://github.com/cloudfoundry/java-buildpack.git#  java-main open-jdk-
1 open-jdk-like-memory-calculator=2.0.1_RELEASE spring-auto-reconfiguration=1.10.0_R

     state     since                    cpu    memory           disk           de
#0   running   2016-05-27 04:37:37 PM   0.0%   381.2M of 512M   152.4M of 1G
#1   running   2016-05-27 04:37:36 PM   0.0%   333.8M of 512M   152.4
DROBERTS-MBPRO:attendee-service droberts$
```

Our application is now running with the 2 instances we specified in the manifest.yml file.