CLOUD FOUNDRY
SUMMIT 2017

OCTOBER 11-12
SWITZERLAND

FREEDOM TO CREATE
An IoT App's Journey to the Cloud
From Localhost to PCF Dev and Pivotal Cloud Foundry

Beibei Yang, Ph.D.
Sr. Advisor

@beibei_yang
The Hype around IoT

“The technology underpinning the Internet of Things (IoT) — acquiring, analyzing, and activating data — is an essential element of generating innovative experiences and transforming operating models.”

- $800 Million in Spending
- 8+ Billion Connected “Things”

2017

Source:
Gartner: Internet of Things — Endpoints and Associated Services, Worldwide, 2016

2021

- $1.4 Trillion in Spending
- 20+ Billion Connected "Things"
Use Case Factory

Reporting environment

Pivotal Cloud Foundry
Native Hybrid Cloud

Pivotal Web Services

Data Center Edge

Public Cloud

Microsoft Azure

Cognitive Services

Data Center

Windows OS & Ubuntu

MQTT

HDFS

VMs

Event Video

mongoDB

Lab 1 Room

Vietnamese

Lab 2 Room

IP cameras

Demo Room

IP cameras

Sensors

MQTT

MQTT

MQTT

MQTT

CLOUD FOUNDARY

SUMMIT 2017
R Shiny Dashboard

Select a Gateway:
SmartEdge Demo

Multiple devices are configured with each gateway

Select Device to monitor:
PowerScout DS 1

Multiple readings are available on each device

Select Device readings to monitor:
absolute_amps_11

Gateway/Device: SmartEdge Demo / PowerScout DS 1 (30 min)

Device Readings
Customized R Buildpack

https://github.com/beibeiyang/cf-buildpack-r
Containers

Friendly to R users

Apps run on both CF and Heroku
Version compatibility issues

Extended staging time

Bulky apps

No-go for libraries using JDK, OpenMP, ...

- RJDBC
- xlsx
- rJava
- arulesNBMiner
- mallet
- RMongo
- OpenNLP
- OpenStreetMap
- RWeka

$ export CF_STAGING_TIMEOUT=60
$ export CF_STARTUP_TIMEOUT=60
Another Direction

• Use an official CF buildpack (such as python_buildpack)
• Run on localhost
• Test in PCF Dev
• Push to NHC PCF / Pivotal Web Services
## Python App in Three Scenarios

### Localhost

<table>
<thead>
<tr>
<th>Language</th>
<th>Python 2.x / Anaconda Python 2.x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libraries</td>
<td>pip install -r requirements.txt</td>
</tr>
<tr>
<td></td>
<td>tornado==4.4.2</td>
</tr>
<tr>
<td></td>
<td>bokeh&gt;=0.12.5</td>
</tr>
<tr>
<td></td>
<td>numpy==1.11.1</td>
</tr>
<tr>
<td></td>
<td>pymongo==3.3.0</td>
</tr>
<tr>
<td></td>
<td>redis==2.10.5</td>
</tr>
<tr>
<td>Start command</td>
<td>bokeh serve \</td>
</tr>
<tr>
<td></td>
<td>--port=8080 \</td>
</tr>
<tr>
<td></td>
<td>--address=localhost \</td>
</tr>
<tr>
<td></td>
<td>--show app</td>
</tr>
<tr>
<td>App URL</td>
<td><a href="http://localhost:8080">http://localhost:8080</a></td>
</tr>
</tbody>
</table>
# Python App in Three Scenarios

<table>
<thead>
<tr>
<th>Language</th>
<th>Python 2.x / Anaconda Python 2.x</th>
<th>PCF Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libraries</td>
<td>pip install -r requirements.txt</td>
<td>Python Built-in Buildpack</td>
</tr>
<tr>
<td></td>
<td>tornado==4.4.2</td>
<td>Python built-in buildpack takes care of library installations</td>
</tr>
<tr>
<td></td>
<td>bokeh&gt;=0.12.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>numpy==1.11.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pymongo==3.3.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>redis==2.10.5</td>
<td></td>
</tr>
<tr>
<td>Endpoint login</td>
<td>NA</td>
<td>cf login -a api.local.pcfdev.io</td>
</tr>
<tr>
<td>Dependencies</td>
<td>Local Redis</td>
<td>cf create-service p-redis</td>
</tr>
<tr>
<td>Start command</td>
<td>bokeh serve --port=8080 --address=localhost --show app</td>
<td>Proefile</td>
</tr>
<tr>
<td>App URL</td>
<td><a href="http://localhost:8080">http://localhost:8080</a></td>
<td>web: bokeh serve --port=8080 --allow-websocket-origin=iotapp1.local.pcfdev.io --address=0.0.0.0 --use-xheaders --log-level=debug app</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Python App in Three Scenarios

<table>
<thead>
<tr>
<th>Localhost</th>
<th>PCF Dev</th>
<th>Pivotal Web Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
<td>Python 2.x / Anaconda Python 2.x</td>
<td>Python Built-in Buildpack</td>
</tr>
<tr>
<td><strong>Libraries</strong></td>
<td>pip install -r requirements.txt</td>
<td></td>
</tr>
<tr>
<td>tornado==4.4.2</td>
<td></td>
<td>Python built-in buildpack</td>
</tr>
<tr>
<td>bokeh&gt;=0.12.5</td>
<td></td>
<td>takes care of library installations</td>
</tr>
<tr>
<td>numpy==1.11.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pymongo==3.3.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>redis==2.10.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start command</strong></td>
<td>bokeh serve \</td>
<td><strong>Procfile</strong></td>
</tr>
<tr>
<td></td>
<td>--port=8080 \</td>
<td>web: bokeh serve --port=8080 \</td>
</tr>
<tr>
<td></td>
<td>--address=localhost \</td>
<td>--allow-websocket- \</td>
</tr>
<tr>
<td></td>
<td>--show app</td>
<td>origin=iotapp1.local.pcfdev.io \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--address=0.0.0.0 \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--use-xheaders \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--log-level=debug app</td>
</tr>
<tr>
<td><strong>Endpoint login</strong></td>
<td>NA</td>
<td>cf login -a api.local.pcfdev.io</td>
</tr>
<tr>
<td><strong>Dependencies</strong></td>
<td>Local Redis</td>
<td>cf create-service p-redis \</td>
</tr>
<tr>
<td></td>
<td>Virtualenv</td>
<td>shared-vm iot_redis</td>
</tr>
</tbody>
</table>
Customer 360

Artificial intelligence (AI)

Customer satisfaction, Productivity, workforce optimization

NOTE: Assume customers opt in to provide personal and locational data.

Scenario

Video Management System detects the license plate of a Highly Valued Customer in the bank’s parking lot then sends an alert to the Banking Staff. The alert will include the mood of the customer from a facial recognition application plus a customer dashboard providing the Banking Staff with important information about the customer. Resulting in high quality customer service.

Acquire and retain customers

Enhanced Customer Engagement

Staffing/Human Capital Management

Image of the customer

Includes:

- Screen containing the Customer’s mood, value, transaction history, etc

Screen containing the Customer’s mood, value, transaction history, etc
Next Steps

- Extended IoT use cases
- Deep Learning
  - GPU as-a-service
- Edge Compute
- Next-gen IoT solution
Appendix
Localhost

- Read sensor data from MongoDB
- Python 2.7 (Anaconda Python 2.7)
- Use virtualenv or conda environments
- Cache data in local Redis
- Dependency libraries
  - Bokeh
  - Tornado
  - Redis
  - pyMongo
  - Numpy

Directory Structure

```
app
  main.py
  set_environ_variables.sh
```
Localhost: Define MongoDB Credentials

- Set up a shell script

```bash
$ cat set_environ_variables.sh
#!/bin/sh
export MONGO_ENV_HOST="mongo.domain.com"
export MONGO_ENV_USERNAME="user"
export MONGO_ENV_PASSWORD="pass"
export MONGO_ENV_AUTHSOURCE="iotlab"
export MONGO_ENV_PORT=27017
```

- Execute `source set_environ_variables.sh`

- In Python script:

```python
mongoip = os.getenv('MONGO_ENV_HOST', mongoip)
mongouser = os.getenv('MONGO_ENV_USERNAME', mongouser)
mongopwd = os.getenv('MONGO_ENV_PASSWORD', mongopwd)
mongoauthsrc = os.getenv('MONGO_ENV_AUTHSOURCE', mongoauthsrc)
mongoport = int(os.getenv('MONGO_ENV_PORT', mongoport))

uri = "mongodb://{}:{}@{}:{}/?authSource={}".format(mongouser, 
mongopwd, mongoip, mongoport, mongoauthsrc)
```
Run on Localhost

- `$ bokeh serve --port=8080 --address=localhost --show app`
- Browse [http://localhost:8080](http://localhost:8080)
Pushing an App

1. Upload app bits and metadata
2. Create and bind services
3. Stage application
4. Deploy application

cf push

push app + app MD

Router

Blobstore

DB

Service credentials

CC + CC Bridge + Brain

Service Broker Node(s)

Cell + Gift = Water
PCF Dev

- Python buildpack takes care of `pip install`

```bash
$ cat requirements.txt
tornado==4.4.2
bokeh>=0.12.5
numpy==1.11.1
pymongo==3.3.0
redis==2.10.5
```

- Procfile

```bash
$ cat Procfile
web: bokeh serve --port=8080 --address=0.0.0.0
   --allow-websocket-origin=iotapp1.local.pcfdev.io
   --use-xheaders --log-level=debug app
```
PCF Dev (Contd.)

• Login via endpoint
  cf login -a https://api.local.pcfdev.io --skip-ssl-validation

• Microservices
  cf create-service p-redis shared-vm iot_redis

• Two ways to define MongoDB Credentials
  • In manifest.yml

    ```yaml
    applications:
    - name: iotapp1
      buildpack: python_buildpack
      memory: 128M
      disk_quota: 256M
      services:
        - iot_redis
      env:
        MONGO_ENV_SERVER_IP: mongo.domain.com
        MONGO_ENV_USERNAME: user
        MONGO_ENV_PASSWORD: pass
        MONGO_ENV_AUTHSOURCE: iotlab
        MONGO_ENV_PORT: 27017
    ```

  • cf cups

    cf cups iotapp-mongo-config -p
    '{"MONGO_ENV_HOST": "mongo.domain.com",
     "MONGO_ENV_USERNAME": "user",
     "MONGO_ENV_PASSWORD": "pass", ...}'
Run on PCF Dev

- cf push
- Microservice bound to app
- https://iotapp1.local.pcfdev.io/
Pivotal Cloud Foundry / PWS

• Python buildpack takes care of `pip install`

```bash
$ cat requirements.txt
tornado==4.4.2
bokeh>=0.12.5
numpy==1.11.1
pymongo==3.3.0
redis==2.10.5
```

• Procfile

```
Procfile for PCF
web: bokeh serve --port=$PORT
   --allow-websocket-origin=iotapp1.cfapps.io
   --address=0.0.0.0 --use-xheaders
   --log-level=debug app

Procfile for PWS
web: bokeh serve --port=$PORT
   --allow-websocket-origin=iotapp1.cfapps.io:4443
   --address=0.0.0.0 --use-xheaders
   --log-level=debug app
```

Directory Structure

```
..cfignore
Procfile
app
   main.py
manifest.yml
requirements.txt
```

https://iotapp1.cfapps.io

https://iotapp1.cfapps.io:4443
Pivotal Cloud Foundry / PWS (Contd.)

• Login via endpoint
  cf login -a api.run.pivotal.io

• Microservices
  cf marketplace
  cf create-service rediscloud 30mb iot_redis

• Two ways to define MongoDB Credentials
  • In manifest.yml
    ```yaml
    ---
    applications:
    - name: iotapp1
      buildpack: python_buildpack
      memory: 128M
      disk_quota: 256M
      services:
        - iot_redis
      env:
        MONGO_ENV_SERVER_IP: mongo.domain.com
        MONGO_ENV_USERNAME: user
        MONGO_ENV_PASSWORD: pass
        MONGO_ENV_AUTHSOURCE: iotlab
        MONGO_ENV_PORT: 27017
    ```
    
  • cf cups
    ```bash
    cf cups iotapp-mongo-config -p
    '{"MONGO_ENV_HOST": "mongo.domain.com",
    "MONGO_ENV_USERNAME": "user",
    "MONGO_ENV_PASSWORD": "pass", ...}'
    ```