Cloudy Middleware

MARK LITTLE <MLITTLE@REDHAT.COM>
TOBIAS KUNZE <TKUNZE@REDHAT.COM>
About

Mark Little
Sr Director of Engineering, Red Hat

Tobias Kunze
PaaS Architect, Red Hat
CTO/Co-founder of Makara
Agenda

1. Why PaaS?
2. The JBoss PaaS
3. OpenShift
4. Demo
WHY PAAS?
Why PaaS?

Development

✔ **Focus** on applications
✖ **Stop** dealing with the stack

Operations

✖ **Stop** dealing with the application
✔ **Focus** on infrastructure

Ease of use

Simplify

Efficiency
PaaS: The “Other” DevOps

PaaS is the smart answer to DevOps’ problem statement

# Administration is Hard Work

## SETUP SERVER

<table>
<thead>
<tr>
<th>Task</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup cloud server</td>
<td>5</td>
</tr>
<tr>
<td>Setup user accounts &amp; keys on cloud server</td>
<td>10</td>
</tr>
<tr>
<td>Install stack</td>
<td>10</td>
</tr>
<tr>
<td>Install stack extensions &amp; libraries</td>
<td>10</td>
</tr>
<tr>
<td>Patch Stack</td>
<td>10</td>
</tr>
</tbody>
</table>

## SCALING

<table>
<thead>
<tr>
<th>Task</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup/configure load balancer</td>
<td>15</td>
</tr>
<tr>
<td>Setup database &amp; replication</td>
<td>30</td>
</tr>
<tr>
<td>Clone application stack to additional server</td>
<td>60</td>
</tr>
</tbody>
</table>

## SECURITY

<table>
<thead>
<tr>
<th>Task</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure security for stack</td>
<td>30</td>
</tr>
<tr>
<td>Configure firewall</td>
<td>10</td>
</tr>
<tr>
<td>Install change audit</td>
<td>30</td>
</tr>
</tbody>
</table>

## MONITORING

<table>
<thead>
<tr>
<th>Task</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install system monitoring</td>
<td>30</td>
</tr>
<tr>
<td>Install/configure application monitoring</td>
<td>10</td>
</tr>
<tr>
<td>Install/configure log aggregation</td>
<td>30</td>
</tr>
<tr>
<td>Setup log indexer service for search</td>
<td>30</td>
</tr>
</tbody>
</table>

**Total:** 5-8h

---

### Set it up once

**5-8hrs**

### Automate it

**25-40hrs**

### Adapt it to change

(diff. release/cloud)

**Ugh!**
Deployment is Hard Work

(51% deployment-related)

PaaS Sweet Spot

Deployments

"Compositeness"

Applications

“Sweet Spot”
Which PaaS?

Development

✔ Focus on applications

✖ Stop dealing with the stack

Ease of use

Operations

✖ Stop dealing with the application

✔ Focus on infrastructure

Efficiency
Which PaaS?

**Development**

✔ **Focus** on applications

✖ **Stop** dealing with the stack

**Ease of use**

**Operations**

✖ **Stop** dealing with the application

✔ **Focus** on infrastructure

**THE RIGHT ABSTRACTION™**

**Simplify**

**Simplify**
PaaS: The Right Abstraction™
Why Cloud?

1. **Cost**: CAPEX, metered
2. **Convenience**: self-service, on-demand
3. **Allure**: elasticity promises scalability
PaaS: The New Linux

Open PaaS

Platform OS

OSS Ecosystem

Platform Kernel

FABRIC

Cloud Management

Cloud Drivers

Cloud
Application Architectures

3 Tiers

Distributed Apps: NoSQL, KV-Stores

Composite Apps
Architectural Styles

- loosely coupled
- monolithic

- transactional
- stateless
What is JBoss PaaS?
Current PaaS approaches

Far too simplistic

- Not standards based
- Little relevance to existing middleware offerings
- Restricts application capabilities
  - So much for dynamic elasticity
  - Existing s/w investments are irrelevant

Limitations are not based on architectural issues

- Security implications

Vendor lock-in dominates
Present and future directions

Build on our existing implementations

We must provide a natural upgrade path for existing users

- We cannot afford to repeat the DCE/CORBA, DCOM/.NET or CORBA/J(2)EE days

Evolution rather than revolution

If the answer is “Cloud 2010” the question is wrong!

Today “Cloud” means “servers”

- More processors outside of “servers” than inside
JBoss PaaS

Users will want to deploy existing applications more dynamically to cloud environments

- Traditional on-premise, standards-based Java
- Want to be able to use existing programming models to deploy traditionally and within cloud environments
- Interoperability between traditional and cloud

Red Hat will create a PaaS offering that will use standard programming technologies to bridge the worlds of on-premise and cloud deployments

- Support the needs of cloud only PaaS opportunities while protecting customers from programming model and API changes
**JBoss PaaS Principles**

Address existing JEE/Java deployment plus cloud

Use existing components of the JBoss portfolio – don’t reinvent the wheel

Don’t invent a closed, proprietary system with new APIs – don’t have to change the programming model

Deploy current JEE/Java on-premise plus cloud deployment

- Make them inter-operable (Future proof investment)
- Deployment environment “agnostic” (elastic)
JBoss PaaS Principles

Developer Centric
- PaaS enable from the viewpoint of the developer
  - CDI
- Integrate tooling for cloud D&D with JBDS
- Make easier for developers to use cloud

Standards Based
- Support PaaS and other cloud standards as they exist
- Define or drive standards where they don’t exist
  - JSR 347
JBoss PaaS Principles

Optimize for Red Hat IaaS offerings and leverage

Market needs addressed by JBoss PaaS:

- Lower costs – greater utilization of resources on demand
- Make deployment easier and more abstracted
- Elasticity to scale on demand
**JBoss PaaS capabilities**

Elastic runtime
- Utilize more compute/storage capacity automatically
- JEE services as true services, not co-located within same VM/Container

Development
- Runtime hosted as a service in cloud (test deploy)
- Develop (tools) locally – then deploy in Cloud

Abstraction to all configuration considerations
- Data store, clustering, network, security services
- Automatic deployment
- Workflow for deployment
Services, VMs and JVMs

Java EE services (in blue) split across machines, VMs and JVMs
Service instance may abstract more cloud instances
- Hierarchical structure of machines, VMs etc.
Similar deployment structure for applications
JBoss PaaS futures

Lifetime application management (tool chain)
- Works across on-premise & clouds
- Supports heterogeneous deployments
- Build time tools
- Deployment tools
- Ongoing operational & management tools

Developer tools (IDE)
- Expansion of developer tools for easier seamless development
How are we going to do it?

Stage 1
- Begin with JBoss EAP and development tools
- Concentrate on the application platform

Stage 2
- Incorporate other “integration” technology for PaaS enablement – ESB, BPM, presentation technologies
- SOA natural architectural approach for PaaS/SaaS
Makara Vision

PROVISION, DEPLOY, MANAGE, MONITOR & SCALE
EXISTING APPLICATIONS
ON CLOUDS
Makara

Built around monitoring

Control, portability, and *visibility*
OpenShift

Two interaction models:

1. OpenShift Express
   - “Runtime-as-a-Service”: simple, git-based interaction

2. OpenShift Flex
   - “Middleware-as-a-Service”
   - Optimized for existing models
     - Nodes
     - Middleware, frameworks, services
     - Software ecosystem
   - Operations use cases: self-serve, user management
Application Overview

These are your application's parameters. You can edit the details here, or work with the application itself on the following pages.

**CLUSTER DNS**
Staging1392187622.stg.rhcloud.com

Running applications can be reached under http://<host>[:port]<context> where:
- `host` is either a local node IP address, the load balancer address (if any), or a configured domain name,
- `port` is the port specified in the static web server configuration (default: 80), and
- `context` is the application's context path (if any).

**DEPLOYMENT HISTORY**
Restore a previous deployment to your development work space.

<table>
<thead>
<tr>
<th>DEPLOYMENT</th>
<th>LAST MODIFIED</th>
<th>COMMENTS</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sun May 1 2011 12:34:44 PM</td>
<td>Onboarded sample app weld-pastecode</td>
<td>RESTORE</td>
</tr>
</tbody>
</table>
Component Selection

Your application depends on some components for support. You’ll need a language runtime and application server.

APPLICATION COMPONENTS

Application Type
- JBoss

Web Server
- Apache HTTP Server (Web Tier) (www-static.apache2.2.3-45.i686.rpm)

Application Server Version
- JBoss Application Server 6.1.0 (jboss-6.0.0_6.0.0-15.all.rpm)

Language Version
- JDK 6.0 (jdk6-1.6.0_1.6.0-11-16.i686.rpm)

Database
- MySQL Server 5.1.52 (mysql-server_5.1.52-8.amd64.rpm)

Other Components
- Zend Framework 1.10.8(zend-framework-1.10.8-1.10.8-2.i686.rpm)
- Memcached 1.4.5(memcached-1.4.5-1.4.5-4.i686.rpm)
- mongoDB-1.6.5(mongodb-1.6.5-23_amd64.rpm)

This application sends email (you must use the Settings tab to configure the cluster to send email)
These are your application's files. If this is the first time you're bringing the application in from outside the OpenShift Flex system then these files should be the same as before the import. Application files may be edited in an ssh session, or uploaded via scp or rsync/ssh by logging in as the username "admin" and the admin password set on cluster creation. Please avoid modifying the "git" directory. With rsync, please use "--exclude git". Location: /home/admin/weld-pastecode/shared/bundle

File Name: bundle/weld-pastecode.war.extracted/display.xhtml

```xml
<link rel="alternate stylesheet" href="syntaxhighlighter/styles/shThemeFadedGray.css" type="text/css" title="FadedGray Theme" media="screen"/>
<link rel="alternate stylesheet" href="syntaxhighlighter/styles/shThemeMidnight.css" type="text/css" title="Midnight Theme" media="screen"/>
<link rel="alternate stylesheet" href="syntaxhighlighter/styles/shThemeDark.css" type="text/css" title="Dark Theme" media="screen"/>
<link rel="alternate stylesheet" href="syntaxhighlighter/styles/shThemeDjango.css" type="text/css" title="Django Theme" media="screen"/>

// define mainarea

<div class="contentHeader">
  Submitted by #pasteWindow.codeFragment.user
  rendered="#(pasteWindow.codeFragment.friendlyDate("just now"))" #pasteWindow.codeFragment.friendlyDate()
</div>

```

© 2011 Red Hat, Inc. All Rights Reserved.
These files configure the components that your application uses to run. For direct file access, ssh as "admin" with the admin password configured when the cluster was created. Location: /home/admin/weld-pastecode/shared/info/setup

Port offset (unique per application):

- Naming: 1099
- Naming RMI: 1098
- Web service: 8083
- Unified invoker connector: 4446
- Invoker JNDI: 4444
- HA-JNDI: 1100
- HA-JNDI RMI: 1101
- HTTP connector: 8080
- HTTPS connector: 8443
- AJP connector: 8009
- HornetQ Netty: 5445
- HornetQ Netty SSL: 5446
- HornetQ Netty batch: 5455
- JBossTS recovery manager: 4712
- JBossTS Transaction Status Manager: 4713
Deployment

Check the deployment summary to see what you’ve changed. When you’re ready to deploy you can choose a deployment mode. If you need to edit anything, return to previous steps.

DEPLOY CHANGES 1 FILES TO 4 CLOUD SERVER(S)

- FILES MODIFIED IN DEVELOPMENT: 1
  - bundle/weld-pastecode.war.extracted/display.xhtml

- CONFIGURATION FILES MODIFIED: 0
  - No files found

- FILES MODIFIED IN PRODUCTION: 0
  - No files found
Transaction Monitoring
### Log Management

#### Table

<table>
<thead>
<tr>
<th>Collection Time</th>
<th>Cloud Server</th>
<th>Application</th>
<th>Component</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>5/1/11 12:46:50</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/16 HTTP/1.1 200 7653</td>
</tr>
<tr>
<td>5/1/11 12:46:50</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/17 HTTP/1.1 200 7345</td>
</tr>
<tr>
<td>5/1/11 12:46:50</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/18 HTTP/1.1 200 7478</td>
</tr>
<tr>
<td>5/1/11 12:46:50</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/19 HTTP/1.1 200 7404</td>
</tr>
<tr>
<td>5/1/11 12:46:51</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/20 HTTP/1.1 200 7994</td>
</tr>
<tr>
<td>5/1/11 12:46:51</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/21 HTTP/1.1 200 8491</td>
</tr>
<tr>
<td>5/1/11 12:46:51</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/22 HTTP/1.1 200 8389</td>
</tr>
<tr>
<td>5/1/11 12:49:57</td>
<td>72.44.52.13</td>
<td>weld-pasteode</td>
<td>Application</td>
<td>INFO [org.jboss.weld.examples.pasteode.session.CodeFragmentPrinterImpl] (pool-22-thread-1) No</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/22 HTTP/1.1 200 8387</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/23 HTTP/1.1 200 7900</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/24 HTTP/1.1 200 7777</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/25 HTTP/1.1 200 8681</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/26 HTTP/1.1 200 9619</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/5 HTTP/1.1 200 7402</td>
</tr>
<tr>
<td>5/1/11 12:52:18</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/1 HTTP/1.1 200 7341</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/2 HTTP/1.1 200 7365</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/3 HTTP/1.1 200 7502</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/4 HTTP/1.1 200 7401</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/5 HTTP/1.1 200 7403</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/6 HTTP/1.1 200 7478</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/7 HTTP/1.1 200 7722</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/8 HTTP/1.1 200 7532</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/9 HTTP/1.1 200 7422</td>
</tr>
<tr>
<td>5/1/11 12:52:19</td>
<td>50.17.3.65</td>
<td>weld-pasteode</td>
<td>Web Server</td>
<td>GET /weld-pasteode/10 HTTP/1.1 200 7704</td>
</tr>
<tr>
<td>Status</td>
<td>Name</td>
<td>Start Time</td>
<td>Elapsed time</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------------------------------------</td>
<td>-------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Updating load balancer: &quot;us-east-1/Staging1592187622&quot;</td>
<td>Sun May 01 12:30:40</td>
<td>0 min 6 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Joining cloud server: &quot;72.44.52.13&quot; to cluster: &quot;Staging&quot;</td>
<td>Sun May 01 12:30:06</td>
<td>0 min 34 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Joining cloud server: &quot;50.17.3.85&quot; to cluster: &quot;Staging&quot;</td>
<td>Sun May 01 12:29:30</td>
<td>0 min 36 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Joining cloud server: &quot;50.16.140.58&quot; to cluster: &quot;Staging&quot;</td>
<td>Sun May 01 12:28:48</td>
<td>0 min 42 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Creating 3 cloud server(s) for cluster with tag: &quot;Staging1392187622&quot;</td>
<td>Sun May 01 12:26:52</td>
<td>1 min 56 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Building cluster: &quot;Staging&quot; on cloud server: &quot;50.17.133.170&quot;</td>
<td>Sun May 01 12:25:10</td>
<td>1 min 41 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Creating load balancer: &quot;Staging1392187622&quot;</td>
<td>Sun May 01 12:25:01</td>
<td>0 min 5 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Creating 1 cloud server(s) for cluster with tag: &quot;Staging1392187622&quot;</td>
<td>Sun May 01 12:24:30</td>
<td>0 min 30 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Creating cluster: &quot;Staging&quot;</td>
<td>Sun May 01 12:24:30</td>
<td>6 min 15 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Authentication: &quot;AWS&quot;</td>
<td>Sun May 01 12:19:48</td>
<td>0 min 6 sec</td>
<td></td>
</tr>
<tr>
<td>Completed</td>
<td>Linking to cloud account: &quot;AWS&quot;</td>
<td>Sun May 01 12:19:48</td>
<td>0 min 7 sec</td>
<td></td>
</tr>
</tbody>
</table>
Auto-Scale Settings

- Enable Auto-Scaling

Minimum Cloud Servers: 2
Maximum Cloud Servers: 20

Scaling Threshold Configuration

Scaling Strategy: Requests per Node

- Scale up if Requests greater than 100 requests / second for 15 minutes (evaluated every 10 minutes)
- Scale down if Requests less than 20 requests / second for 30 minutes (evaluated every 20 minutes)

CANCEL  SAVE
## Server Monitoring

### Server List

<table>
<thead>
<tr>
<th>NAME</th>
<th>CLOUD</th>
<th>CLUSTER</th>
<th>VIRTUAL IP</th>
<th>ADMIN</th>
<th>SERVING</th>
<th>STATUS</th>
<th>RESOURCES (10 MIN AVERAGE)</th>
<th>ACTIVITY</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>50.17.133.170</td>
<td>AWS</td>
<td>Staging</td>
<td>No</td>
<td></td>
<td>✔️</td>
<td>Synched</td>
<td><img src="image" alt="CPU, Memory, Storage, TX Bytes, RX Bytes Graph" /></td>
<td><img src="image" alt="Activity Graph" /></td>
<td>UNJOIN</td>
</tr>
<tr>
<td>50.16.140.58</td>
<td>AWS</td>
<td>Staging</td>
<td>No</td>
<td></td>
<td>✔️</td>
<td>Synched</td>
<td><img src="image" alt="CPU, Memory, Storage, TX Bytes, RX Bytes Graph" /></td>
<td><img src="image" alt="Activity Graph" /></td>
<td>RESTART</td>
</tr>
<tr>
<td>50.17.3.85</td>
<td>AWS</td>
<td>Staging</td>
<td>No</td>
<td></td>
<td>✔️</td>
<td>Synched</td>
<td><img src="image" alt="CPU, Memory, Storage, TX Bytes, RX Bytes Graph" /></td>
<td><img src="image" alt="Activity Graph" /></td>
<td>UNJOIN</td>
</tr>
<tr>
<td>72.44.52.13</td>
<td>AWS</td>
<td>Staging</td>
<td>No</td>
<td></td>
<td>✔️</td>
<td>Synched</td>
<td><img src="image" alt="CPU, Memory, Storage, TX Bytes, RX Bytes Graph" /></td>
<td><img src="image" alt="Activity Graph" /></td>
<td>RESTART</td>
</tr>
</tbody>
</table>
INTRODUCTION

BEGINNING TASKS

Welcome to OpenShift. During signup, you configured a cloud provider, allocated cloud servers into a server cluster, and deployed a sample application. Here you can re-do those steps but why not move on to other guides? Each guide will start with a few explanatory screenshots and then navigate you to the right page to get started.

- DEFINE A CLOUD ACCOUNT
  You already setup your cloud account during signup but if you’d like to use an additional cloud account or re-enter your cloud credentials, start here.

- CREATE A SERVER CLUSTER
  During signup, you created a server cluster. Come back here if you’ve deleted it and want to create one again, or if you want to create additional server clusters. Multiple applications can share a server cluster (and thus a URL space and scaling policies) or you can create a server cluster for each application.

- DEPLOY SAMPLE APPLICATION
  During signup, you deployed an application from the library of samples. If you’ve deleted it, or want to deploy another one, this guide will get you started. Keep in mind that running multiple applications on the same cluster requires more resources... usually RAM is the constraining resource.

- VIEW RUNNING APPLICATION
  Find the URL to use to exercise the running application

- RECONFIGURE SAMPLE APPLICATION
  Learn a little about application deployment definitions by making a simple change to the application’s URL routing configuration

- SCALE SAMPLE APPLICATION
  Scale up to handle more load, or scale down to save money

- MONITOR APPLICATION PERFORMANCE
  Monitor the performance of your application.

- DEFINE AND DEPLOY YOUR OWN APPLICATION
  Now that you’ve used a sample application, upload your own application to get deployed in the cloud

RESOURCES

HOW-TO GUIDES

- Getting Started with OpenShift Flex
- Deploying JBoss with OpenShift Flex
- Deploying Drupal with OpenShift Flex
Q&A

THANKS!