IMPORT CONTINUOUS.DELIVERY.*

Anton Arhipov
@antonarhipov

Product Lead
ZeroTurnaround
AGENDA

• Continuous Delivery Pipeline
• Tools
• Demo
Have you signed up for myJRebel yet?

Learn about JRebel

Eliminate application restarts in Java with JRebel! JRebel is a JVM plugin that eliminates application redeployments from the Java development cycle, a process that takes over 10 minutes of coding time away from developers each working hour, according to a recent survey. Just code, refresh and see everything instantly.

Learn More
Before CD

- Developer
- QA
- Release Engineer
- SysOp
Powered by CD

Developer → QA → Manager

CD Engineer

SysOp
Fedex process

- Package
- Dropoff
- Transfer
- Delivery
- Profit!
Java EE

- Package
- Test
- Approve
- Deploy
- Profit!
Questions?

- What exactly is in prod now?
- How did it get deployed?
- How did it get there?
- Where did it come from?
- How did you package the application?
Fedex FAIL
Software FAIL

We're sorry, but your Gmail account is temporarily unavailable. We apologize for the inconvenience and suggest trying again in a few minutes.

If the issue persists, please visit the Gmail Help Center.

Try Again  Sign Out

Hide Detailed Technical Info
Numeric Code: 21

©2008 Google - Gmail Home - Privacy Policy - Program Policies - Terms of Use - Google Home
Continuous Delivery Pipeline
A PIPELINE?

Source: http://startupblog.files.wordpress.com/2008/09/pipeline1.jpg
Continuous Delivery pipeline!

Source: http://www.flickr.com/photos/mikeleishman/1787764289/
The Pipeline

- Orchestration Platform
- Delivery Manager
- Artifact Repository

Let's talk about tools!
JENKINS
(OSS CONTINUOUS INTEGRATION SERVER)
Orchestration
Platform
LiveRebel
(Delivery Manager)
What can you do with LiveRebel?

- **Deploy/Undeploy**
- **Update**
  - **No downtime**
  - **No lost sessions**
- **Plain old restarts**
- **Environment/deployment status**
ARTIFACTORY

(BINARY REPOSITORY)
The Pipeline / The Repos
PIPELINE PHASES
**Build Phase**

- **store artifacts**
  - artifactory

- **exec processes**
  - Jenkins
    - pull down code
    - build WAR with Maven
    - generate trace
    - upload WAR and trace
    - trigger "deploy-test" job

- **deploy apps**
  - LiveRebel

- **run apps**
  -...
Test phase

- **store artifacts**
  - Artifactory

- **exec processes**
  - Jenkins
    - download WAR and trace
    - deploy app and trace

- **deploy apps**
  - LiveRebel
    - deploy app

- **run apps**
  - test env

- **"deploy-test" job**
  - "build" repo

- **"automatic-tests" job**
  - "build" repo
  - test and generate trace
  - upload WAR and trace
  - trigger "begin-qa" job
### QA Phase

<table>
<thead>
<tr>
<th>Store Artifacts</th>
<th>&quot;begin-qa&quot; job</th>
<th>&quot;qa-success&quot; job</th>
<th>&quot;clean-up-test&quot; job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artifactory</td>
<td>&quot;test&quot; repo</td>
<td>&quot;test&quot; repo</td>
<td>&quot;QA&quot; repo</td>
</tr>
</tbody>
</table>

**Exec Processes**

- **Jenkins**
  - Download trace
  - Send e-mail
  - Download WAR and trace
  - Generate trace
  - Upload WAR and trace
  - Remove app
  - Trigger "begin-deploy-production" job
  - Undeploy app

**Deploy Apps**

- LiveRebel

**Run Apps**

- Test env
# Production Phase

<table>
<thead>
<tr>
<th>Production Phase</th>
<th>&quot;begin-deploy-production&quot; job</th>
<th>&quot;deploy-production&quot; job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store artifacts</td>
<td>&quot;QA&quot; repo</td>
<td>&quot;RC&quot; repo</td>
</tr>
<tr>
<td></td>
<td>artifactory</td>
<td></td>
</tr>
<tr>
<td>Execute processes</td>
<td>Jenkins</td>
<td></td>
</tr>
<tr>
<td></td>
<td>download WAR and trace</td>
<td>upload WAR and trace</td>
</tr>
<tr>
<td></td>
<td>generate trace</td>
<td>send e-mail</td>
</tr>
<tr>
<td>Deploy apps</td>
<td>LiveRebel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>deploy app</td>
<td></td>
</tr>
<tr>
<td>Run apps</td>
<td>production env</td>
<td></td>
</tr>
</tbody>
</table>
Dirty details in Jenkins
Pipeline view in Jenkins
Step configuration in Jenkins

```
Execute shell

See the list of available environment variables
```

Deploy or Update artifact with LiveRebel

Choose your action: Deploy or update
Artifact: demo-chat.war

- Override the version information in the artifact
Trace file:
Context path: demo-chat-${SOURCE_BUILD_NUMBER}
Servers:
- prod
- tomcat-1 (offline)
- tomcat-2 (offline)

Show advanced Update settings
Update Strategies
curl -X POST
-H "Content-Type: application/vnd.org.jfrog.artifactory.build.PromotionRequest+json"
-d {
"status": "in qa",
"comment" : "Promoted to QA",
"ciUser": "jenkins",
"targetRepo" : "qa"}

http://host:8081/artifactory/api/build/promote/app/${BLD}
Package status information
Themes Not Covered

• Database
• Configuration & Environment
• Tests & Monitoring
• Rollback
Questions?

• How do you package the application?
• Where did it come from?
• Where does it go?
• How does it get deployed?
• What exactly is in prod now?
The answers are in the pipeline!
Pipeline Summary

- **Jenkins jobs represent the workflow**
- **Artifactory is a sync-point**
- **LiveRebel manages apps and versions**
- **Manual flows with REST**
- **Tracking with scripts & text files**
WHAT'S CHANGED?
What’s changed?

• Increased significance of automated QA
What’s changed?

- Increased significance of automated QA
- Reduced manual testing
What’s changed?

- Increased significance of automated QA
- Reduced manual testing
- Infrastructure roles NOT involved with release
What’s changed?

- **Increased significance of automated QA**
- **Reduced manual testing**
- **Infrastructure roles NOT involved with release**
- **Release approval is a business decision**
What’s changed?

- **Increased significance of automated QA**
- **Reduced manual testing**
- **Infrastructure roles NOT involved with release**
- **Release approval is a business decision**
- **Automation is the key**

Sounds simple, eh?
Questions?

@antonarhipov
anton@zeroturnaround.com