Clustering Quartz

Ryan Vanderwerf

Chief Architect

ReachForce

www.reachforce.com

My Background

- Currently building a Grails and Cloud based infrastructure for ReachForce
- Architected a Grails solution for Developerprogram.com that allows rapid deployment of Developer Program portals for all kinds of companies, specializing in the mobile industry.
- Built Java and Linux based webcasting for events such as SXSW, built telecom software, and ASP's for the financial sector
- Worked with Java since 1996, and built server-side applications ever since
- Enticed into the Groovy and Grails space by speakers at the early NFJS conferences

Clustering Quartz

What Will We Cover?

- Quartz Basics
- Reasons for clustering
- JDBCJobStore clustering
- Terracotta setup and clustering
- Quartz Plugin
- Quartz2 Plugin
- Demo applications of both

Clustering Quartz

What Is Quartz?

- Open source Java API Used to schedule, persist, and distribute jobs
- Great Grails support
- Great community support and large usage (thousands)
- Most common solution for scheduling execution in Java applications

Clustering Quartz

Quartz Plugin

- Integrates Grails with Quartz 1.x
- Works best with Clustered Terracotta Option due to bug in Terracotta (https://jira.terracotta.org/jira/browse/QTZ-310)
- Yet to be updated to support Quartz 2
- Officially supported by SpringSource

Clustering Quartz

Quartz2 Plugin

- Integrates Grails with Quartz 2.x
- Works best with Clustered JDBCStore due to bug in Terracotta (https://jira.terracotta.org/jira/browse/QTZ-310)
- Supports Groovy based JobDetail
- Not Officially supported by SpringSource
- Supports (nosql) engines like Mongo or Redis

Clustering Quartz

Why Cluster Quartz

- Distribute Load
- Scale easily
- Handle many batch jobs at once
- Persist scheduled work queue in case of crash
- Fail-over

Clustering Quartz

Installing Quartz and creating a job

- grails install-plugin quartz
- grails create-job <jobName>

```
class MyJob {
  static triggers = {
    simple name: 'mySimpleTrigger', startDelay:
60000, repeatInterval: 1000
  }
  def group = "MyGroup"
  def execute(){
    print "Job run!"
  }
}
```

Clustering Quartz

Installing Quartz and creating a job

- Cron trigger example

```
class MyJob {
  static triggers = {
    cron name: 'myTrigger', cronExpression: "0 0 6 * * ?"
  }
  def group = "MyGroup"
  def execute(){
    print "Job run!"
  }
}
```

Clustering Quartz

Installing Quartz and creating a job

- Dynamic Jobs

```
// creates cron trigger;
MyJob.schedule(String cronExpression, Map params?)
//  creates simple trigger: repeats job repeatCount+1 times with delay of
repeatInterval milliseconds;
MyJob.schedule(Long repeatInterval, Integer repeatCount?, Map
params?) )
// schedules one job execution to the specific date;
MyJob.schedule(Date scheduleDate, Map params?)
//schedules job's execution with a custom trigger;
MyJob.schedule(Trigger trigger)
// force immediate execution of the job.
MyJob.triggerNow(Map params?)

// Each method (except the one for custom trigger) takes optional
'params' argument.
// You can use it to pass some data to your job and then access it from
the job:
class MyJob {
  def execute(context) {
    println context.mergedJobDataMap.get('foo')
  }
}
// now in your controller (or service, or something else):

MyJob.triggerNow([foo:"It Works!"])
```

Clustering Quartz

Prepping your environment for distributed quartz

- How do you want to run the jobs?
  - Replicate your grails app X times and distribute across that?
  - Have separate replicated application the picks up the jobs
    - WAR file under app server
    - Standalone application
      - Just Java classes running scheduler command line?
      - Standalone plugin?
      - Custom standalone i.e. https://gist.github.com/1804182 ?

Clustering Quartz

Terracotta vs. JDBCJobStore Clustering

- Terracotta has remote GUI console
- Terracotta shows quick status of jobs
- Terracotta doesn't require a database to persist jobs
- Terracotta Open Source Free
- Good support via Terracotta.org forums
- Both solutions should have a time server synchronizing clocks on all machines

Clustering Quartz

JDBCJobStore Clustering

- Most common and least work to configure
- Set up JobStore and Delegate
- Configure database table prefix
- Setup database schema
- Works with Grails 'quartz' (Quartz 1.8) or 'quartz2' (Quartz 2) plugin

Clustering Quartz

Terracotta Clustering

- No Database setup required
- Currently only works with Quartz 1.8 and 'quartz' plugin
- Doesn't work with Quartz2 plugin because it implements a different class for the JobDetails interface that is not JobDetailsImpl. Terracotta will throw errors because it assume JosDetailsImpl class is used. JIRA logged at https://jira.terracotta.org/jira/browse/QTZ-310 if you'd like to vote on it
- Slick GUI Interface
- Excellent failover and hot spare ability, as well as integrating with Hibernate 2nd level cache, and HTTP Session caching
- Open source edition free, works great for most small to medium installs
- Commercial version allows more than one active cluster server and role based access to admin console

Clustering Quartz

Setting Up Open Source Terracotta Clustering

- Download Terracotta 3.4.1 from (This is the last version that is compatible with Quartz 1.8.x)
- Install 'quartz' grails plugin
- Create quartz.properties in grails-app/conf or src/java

Clustering Quartz

Setting Up Open Source Terracotta Clustering

- Sample quartz.properties file:
  - org.quartz.scheduler.instanceName = MyClusteredScheduler
  - org.quartz.scheduler.instanceId = AUTO
  -
  - #===
    =========================
  - # Configure ThreadPool
  - #===
    =========================
  -
  - org.quartz.threadPool.class = org.quartz.simpl.SimpleThreadPool
  - org.quartz.threadPool.threadCount = 25
  - org.quartz.threadPool.threadPriority = 5
  - org.quartz.jobStore.class=org.terracotta.quartz.TerracottaJobStore
  - # the path below should point to your terracotta config file. This can can be a URL as well like http://
  - org.quartz.jobStore.tcConfigUrl = /opt/terracotta-3.4.1/tc-config.xml

Clustering Quartz

Starting Terracotta

- Download Terracotta 3.4.1 from

  http://terracotta.org/downloads/open-source/destination?name=te

- Copy $TERRACOTTA_HOME/config-samples/tc-config-express-reference.xml to $TERRACOTTA_HOME/tc-config.xml
- Run ./start-tc-server.sh -f /path/to/tc-config.xml
- Run ./dev-console.sh

Clustering Quartz

Terracotta Demo

- 2 Nodes CreatePersonJob

Clustering Quartz

JDBCJobStore Clustering

- Simple to configure
- Schema to create tables included in distibution
- Uses centeral database to persist jobs, and unique nodeIDs (AUTO will work)
- Time server must sync instances regularly, and be within 1 second of each other
- Works with Quartz 1.8 or Quartz 2.0 ('quartz' or 'quartz2' plugin

Clustering Quartz

JDBCJobStore Clustering

- Sample quartz config highlights (see code samples for complete file)
  - org.quartz.jobStore.isClustered = true
  - org.quartz.jobStore.clusterCheckinInterval = 20000
  - org.quartz.jobStore.misfireThreshold = 60000
  - org.quartz.jobStore.class = org.quartz.impl.jdbcjobstore.JobStoreTX
  - org.quartz.jobStore.driverDelegateClass = org.quartz.impl.jdbcjobstore.oracle.OracleDelegate

Clustering Quartz

JDBCJobStore Demo

- 2 Nodes CreatePersonJob with JDBCJobStore

Clustering Quartz

Go Advanced!

- Use Job and Trigger Listeners (there is no groovy version of this, you will have to make regular class files) to split large jobs apart into smaller ones.
- Example applications using distributed jobs:
  - Email Campaign Tool
  - Data processing
  - Email Verification Tool

Clustering Quartz

Setup and run Quartz 2 plugin

- Use Job and Trigger Listeners (there is no groovy version of this, you will have to make regular class files) to split large jobs apart into smaller ones.
- See Docs at https://github.com/9ci/grails-quartz2

Clustering Quartz

Setup and run Quartz 2 plugin

- Use Job and Trigger Listeners (there is no groovy version of this, you will have to make regular class files) to split large jobs apart into smaller ones.
- See Docs at https://github.com/9ci/grails-quartz2
- Example applications using distributed jobs:
  - Email Campaign Tool
  - Data processing
  - Email Verification Tool

More Information

https://github.com/9ci/grails-quartz2
http://terracotta.org/downloads/open-source/catalog
https://jira.terracotta.org/jira/browse/QTZ-310
http://grails.org/plugin/quartz
http://grails-plugins.github.com/grails-quartz/
http://quartz-scheduler.org/

Contact Me

Via twitter: https://twitter.com/RyanVanderwerf
Google+/email: rvanderwerf@gmail.com
Blog: http://rvanderwerf.blogspot.com