JobQueue development environment setup

Tuesday, February 07, 2017 7:41 AM

Job Queue is actually now a "misnomer": it does not actually queue up jobs -- that was planned, but was canceled. The reason for the cancelation was a) concern for high chance of race conditions due to inability to sync with STAF Cron processes/operations, the latency for updating a job is too high (several seconds). It fell back to weekly batch prioritization.

Resources

- GIT Repo: https://gitlab.factset.com/app-qa-automation/jobqueue
- JobQueue Status: http://tellusstgb01.pc.factset.com:8080/JobQueueEJB/api/status
- Deployment Location: \\tellusstgb01\apache-tomee-plus-1.7.3\webapps\

Dependencies

- JDBC SQL Server Driver: https://www.microsoft.com/en-us/download/details.aspx?id=11774
  Copy its jar file to TomEE\lib\ (it is already done)
- You need to install TomEE 1.7.4 on your development laptop:
  http://tomee.apache.org/downloads.html
  Choose TomEE Plus version 1.7.4.

Compatibility

- Compatible with TomEE Plus 1.7.3 and 1.7.4 (Tomcat v7.0)
- TomEE Plus 7: It works, but some dependencies in pom.xml does not work as-is, so you would need to comment them out, and add TomEE jar files to project Build Path instead.

Development Environment Setup (Eclipse)

- Download source code:

  ```
  git clone git@github.factset.com:app-qa-automation/jobqueue.git
  ```

  The local repo will be created in the current directory.

- You should develop in your own branch:

  ```
  git branch branchname
  git checkout branchname
  ```

  Substitute *branchname* by your own branch name.

  Later when it is ready for release, then you can merge into master:

  ```
  git checkout master
  git merge branchname
  ```

- Create project in Eclipse:
  - Select **Maven > Import Maven > Existing Maven project**
  - Select JobQueue project directory above
- Add server runtime:
  - Select **Window > Preferences**
  - Select **Server > Runtime Environments > Add**
  - Select **Apache > Apache Tomcat v7.0**

    - Click **Next**
    - Select your local TomEE directory, for example `C:\apache-tomee-plus-1.7.4`
    - Click **Finish**
- Runtime environment setup:
    - Right-click project and select **Properties**
    - Select **Targeted Runtimes**
    - Check **Apache Tomcat v7.0** created above
    - Click **OK**
- Add SQL Server JDBC driver dependency:
    - Right-click project and select **Build Path > Configure Build Path**
    - Click **Add External Jars**
    - Select path to **sqljdbc42.jar**
    - Click **OK**

Building the Project

Eclipse automatically compiles when you modify the source code, so no need to build explicitly.

Running the Web Service in Eclipse

- Right click JobQueueEJB project
- Select Run As > Run on Server or Debug As > Debug on Server

  Note: It is presently not configured to run as Maven operations (such as build or test).

Unit Tests

Unit tests are under JobQueueEJB (project) > Java Resources > src/test/java > com.factset.appqaauto.jobqueue.test.

Unit tests for Prioritization ("Leveling jobs"):

Comment out any unit test that you don't want to run
Right-click LevelJobsTest.java in Project Explorer or editor
Select Debug As > Junit Test, or Run As > Junit Test
The unit test `testLevelSpecifiedDate` prioritizes all jobs of a specific day.

Unit tests for JEP requests queue:

SchedulerQueueTest.java

Deployment

To package a .war for deployment:

- Right-click project
- Select **Export > WAR file**
- Select any **Destination** folder (use a local directory for performance)
- Click **Finish**

Installation:

- Copy the .war file to *TOMEE*/**webapps**/ directory on tellusstgb01 (there is Windows Share).
- It should automatically be reloaded by TomEE, but sometime the state might be messed up, so restart TomEE to start with clean state

TomEE:

- TomEE directory is C:\apache-tomee-plus-1.7.3 on tellusstgb01
- TomEE runs as Windows Service called "Apache TomEE"

Configuration

C:\Tellus\JobQueue\JobQueueConfig.properties

Example:

```
# debug = true
# debugDelay = 5000

logFile = C:/Tellus/JobQueue/logs/JobQueue.log
logLevel = ALL
# Max bytes per log file in MB
logLimit = 10
# Max number of log files in rotation
logCount = 15

# Leveling (prioritization) time
levelHour = 0
levelMinute = 30
#levelSecond = 0

priorityScheduler Url =
http://tellusstgb01.pc.factset.com:8080/RestfulSchedulingService/scheduler/jobQueueRequest
schedulerTaskUrl =
http://tellusstgb01.pc.factset.com:8080/RestfulSchedulingService/scheduler/schedulejobs

dbPrimaryServer = tellus-sql-primary.prod.factset.com
dbFailoverServer = tellus-sql-failover.prod.factset.com
dbIntegratedAuthentication = true

# dbUser and dbPassword take effect only if dbIntegratedAuthentication is true
# Backslash must be escaped as double-backslash (\\)
#dbUser = GREENWICH\\svc-tellus
#dbPassword =

# loopInterval msec
loopInterval = 30000

# tellusServers: comma-separated list of servers; do not use domain (will
automatically append it)
# <<ALL>> or omission means all servers
tellusServers = TELLUSB01, TELLUSDEVB01
cronServers = TELLUSB01, TELLUSDEVB01

# extra time for tardy job to finish, as int percentage
# @deprecated
tardyExtraPercent = 25
```

Note: The levelHour and levelMinute schedules jobqueue to prioritize next day at 00:30 AM every day. It will prioritize only the hosts specified by tellusServers and cronServers.

Logs

Log location is configured by logFile in C:\Tellus\JobQueue\JobQueueConfig.properties:
`C:/Tellus/JobQueue/logs/JobQueue.log`

Restart

To restart JobQueue, simply restart TomEE Windows Server on tellusstgb01.

JobQueue code overview

Wednesday, June 07, 2017 10:11 AM

- JobQueue is RESTful Web Service implemented with Java EJB.
- It is currently compatible with TomEE plus 1.7.3.
- It uses Maven for build tool and dependency management.

- Application code is under src/main/java/.
- **Unit tests** are under src/test/java/.
- **Main class** is com.factset.appqaauto.jobqueue.JobQueueBean.
- JobQueueConfig is **configuration** object.

- **REST entry point** is defined by @GET and @POST methods in JobQueueBean class
- /status REST entry point:

```
@GET
@Path("/status")
@Lock(LockType.READ)
public Response status() {
...
```

- The prioritization REST entry point that can be used to manually prioritize a day is:

```
@POST
@Path("/level")
@Consumes(MediaType.APPLICATION_JSON)
@Lock(LockType.READ)
public Response level(LevelRequest request) {
...
```

  Note: That entrypoint is not used by JobQueue for daily prioritizations, but only for admin's manual operation.

  - It will move jobs in DB and Cron, if the scheduler returns them with new start time.
  - It will also save the resourcenumber (which is the number for FPE VM or SGE browser+version combination) of the jobs from the scheduler response.

- Scheduler request queuing: This is REST entry point for queuing requests from JEP:

```
@POST
@Path("/schedulejobs")
@Consumes(MediaType.APPLICATION_JSON)
@Lock(LockType.READ)
public Response scheduleJobs(String request) {
...
```

  - SchedulerQueue class is the main logic for queuing requests from JEP.
  - It forwards requests to scheduler (RestfulSchedulingService).
  - If JEP request has no GroupID (as in the case of coming from browser), then JobQueue passes through the request to JobScheduler without processing (no need for any processing such as moving jobs)
  - Data format is designed to be identical to and compatible with JobScheduler

JobQueue REST API

Friday, February 12, 2016 11:27 PM

Usage

/api/status

Get overall status
GET http://localhost:8080/JobQueueEJB/api/status
Response:

```
{
 "message": "OK",
 "status": "OK"
}
```

/api/debugStatus

Get debugging info
GET http://localhost:8080/JobQueueEJB/api/debugStatus
Response:

```
{
 "busy": false,
 "message": "Timed out waiting for response requestID=46",
 "execQueue": [],
 "status": "OK",
 "responseMessages": {}
}
```

Note: If RestfulSchedulingService does not response within a timeout period, JobQueue will timeout and show the "Tim out" message in the example response.

/api/level

Manually prioritizing a particular date.
POST http://localhost:8080/JobQueueEJB/api/level
Content-Type: application/json
Format:

```
{
  "level-request":
  {
    "start": "2017-06-10"
    "end": "2017-06-11"
  }
}
```

Example:

```
{
  "level-request":
  {
    "start": "2017-06-10"
  }
}
```

The end field is optional; default value is start+1 day (exclusive, i.e. range does not include end).

/api/schedulejobs

Queuing JEP requests and forwards to RestfulSchedulingService.
This is designed to be exactly "plug-in compatible" with RestfulSchedulingService's request/response formats.

- If request has ***no groupid***, then it will pass through to RestfulSchedulingService without further processing and return the scheduler response, as is, to the caller (will not move jobs that changed start time). That is usually from the JEP browser UI, which is temporary start time information and does not need to move any job.
- If request ***has groupid***, then it will try to move any job returned by scheduler that changed start time. And return the original jobs from the request but with new start time. This is usually from JEP PHP that does need to move jobs that scheduler changed start time.

Request format:
POST http://tellusstgb01.pc.factset.com:8080/JobQueueEJB/api/schedulejobs
Content-Type: application/json

Request data format:

```
{
 "task": [
  {
   "start": "2017-06-07T23:00:00",
   "end": "2017-06-07T23:12:00",
   "priority": 4,
   "testenvironment": "FPE",
   "sgbrowsername": null,
   "sgbrowserversion": null
  }
 ]
}
```

Example request with no groupid:

```
{
  "task": [
    {
      "start": "2017-06-07T23:00:00",
      "end": "2017-06-07T23:12:00",
      "priority": 4,
      "testenvironment": "FPE",
      "sgbrowsername": null,
      "sgbrowserversion": null
    }
  ]
}
```

Example response:

```
{
  "updatedJobs": {
    "updatedjobs": [
      {
        "eventid": 0,
        "cronid": 0,
        "durationinmins": 20,
        "starttime": "23:00:00.0000000",
        "startdate": "2017-06-07",
        "isresult": false,
```

```
            "isdev": 0,
            "sgbrowserversion": "",
            "end": "2017-06-07T23:20:00.0000000",
            "id": 0,
            "fixedtime": 0,
            "oldend": "2017-06-07 23:20:00.000",
            "numoftestcases": 0,
            "groupid": 0,
            "start": "2017-06-07T23:00:00.0000000",
            "endtime": "23:20:00.0000000",
            "oldstart": "2017-06-07 23:00:00.000",
            "testenvironment": "FPE",
            "priority": 4,
            "numberofvirtualmachines": 1,
            "seriesid": 0,
            "resourcenumber": 51,
            "allday": false,
            "sgbrowsername": "",
            "enddate": "2017-06-07",
            "precedent": 0,
            "lastmodified": 0,
            "starteditable": false,
            "isexception": 0
          }
        ]
      }
    }
```


TimelineTest (test/admin tool)

http://tellusb01.pc.factset.com/autocrontest/timelineTest.html

This is tool to help visualize the arrayngement of jobs.

JobQueue development environment setup

Tuesday, February 07, 2017 7:41 AM

- GIT Repo: https://gitlab.factset.com/app-qa-automation/jobqueue
- JobQueue Status: http://tellusstgb01.pc.factset.com:8080/JobQueueEJB/api/status
- Deployment Location: \\tellusstgb01\apache-tomee-plus-1.7.3\webapps\

Dependencies

- JDBC SQL Server Driver: https://www.microsoft.com/en-us/download/details.aspx?id=11774
  Copy its jar file to TomEE\lib\ (it is already done)
- You need to install TomEE 1.7.4 on your development laptop:
  http://tomee.apache.org/downloads.html
  Choose TomEE Plus version 1.7.4.

Compatibility

- Compatible with TomEE Plus 1.7.3 and 1.7.4 (Tomcat v7.0)
- TomEE Plus 7: It works, but some dependencies in pom.xml does not work as-is, so you would need to comment them out, and add TomEE jar files to project Build Path instead.

Eclipse Setup

- Download source code:

  ```
  git clone git@gitlab.factset.com:app-qa-automation/jobqueue.git
  ```

  The local repo will be created in the current directory.

- You should develop in your own branch:

  ```
  git branch branchname
  git checkout branchname
  ```

  Substitute *branchname* by your own branch name.

  Later when it is ready for release, then you can merge into master:

  ```
  git checkout master
  git merge branchname
  ```

- Creating project in Eclipse:
  - Select **File > Import**
  - Select **Maven > Existing Maven project**
  - Click **Next**
  - Select JobQueue project directory above
  - Click **Finish**
- Add server runtime:
  - Select **Window > Preferences**
  - Select **Server > Runtime Environments > Add**
  - Select **Apache > Apache Tomcat v7.0**
  - Click **Next**
  - Select your local TomEE directory, for example `C:\apache-tomee-plus-1.7.4`

    - Click **Finish**
- Runtime environment setup:
    - Right-click project and select **Properties**
    - Select **Targeted Runtimes**
    - Check **Apache Tomcat v7.0** created above
    - Click **OK**
- Add SQL Server JDBC driver dependency:
    - Right-click project and select **Build Path > Configure Build Path**
    - Click **Add External Jars**
    - Select path to **sqljdbc42.jar**
    - Click **OK**

Building the Project

Eclipse automatically compiles when you modify the source code, so no need to build explicitly.

Running the Web Service in Eclipse

- Right click JobQueueEJB project
- Select Run As > Run on Server or Debug As > Debug on Server

  Note: It is presently not configured to run as Maven operations (such as build or test).

Unit Tests

Unit tests are under JobQueueEJB (project) > Java Resources > src/test/java > com.factset.appqaauto.jobqueue.test.

Unit tests for Prioritization ("Leveling jobs"):

Comment out any unit test that you don't want to run
Right-click LevelJobsTest.java in Project Explorer or editor
Select Debug As > Junit Test, or Run As > Junit Test
The unit test `testLevelSpecifiedDate` prioritizes all jobs of a specific day.

Unit tests for JEP requests queue:

SchedulerQueueTest.java

Deployment

Maven creating .war file:

```
mvn package -DskipTests
```

Eclipse: package a .war for deployment:

- Right-click project
- Select **Export > WAR file**
- Select any **Destination** folder (use a local directory for performance)
- Click **Finish**

Installation:

- Copy the .war file to *TOMEE*/**webapps**/ directory on tellusstgb01 (there is Windows Share).
- It should automatically be reloaded by TomEE, but sometime the state might be messed up, so restart TomEE to start with clean state

TomEE:

- TomEE directory is C:\apache-tomee-plus-1.7.3 on tellusstgb01
- TomEE runs as Windows Service called "Apache TomEE"

Configuration

C:\Tellus\JobQueue\JobQueueConfig.properties

Example:

```
# debug = true
# debugDelay = 5000

logFile = C:/Tellus/JobQueue/logs/JobQueue.log
logLevel = ALL
# Max bytes per log file in MB
logLimit = 10
# Max number of log files in rotation
logCount = 15

# Leveling (prioritization) time
levelHour = 0
levelMinute = 30
#levelSecond = 0

priorityScheduler Url =
http://tellusstgb01.pc.factset.com:8080/RestfulSchedulingService/scheduler/jobQueueRequest
schedulerTaskUrl =
http://tellusstgb01.pc.factset.com:8080/RestfulSchedulingService/scheduler/schedulejobs

dbPrimaryServer = tellus-sql-primary.prod.factset.com
dbFailoverServer = tellus-sql-failover.prod.factset.com
dbIntegratedAuthentication = true

# dbUser and dbPassword take effect only if
dbIntegratedAuthentication is true
# Backslash must be escaped as double-backslash (\\)
#dbUser = GREENWICH\\svc-tellus
#dbPassword =

# loopInterval msec
loopInterval = 30000

# tellusServers: comma-separated list of servers; do not use domain
(will automatically append it)
# <<ALL>> or omission means all servers
tellusServers = TELLUSB01, TELLUSDEVB01
cronServers = TELLUSB01, TELLUSDEVB01

# extra time for tardy job to finish, as int percentage
# @deprecated
tardyExtraPercent = 25
```

Note: The levelHour and levelMinute schedules jobqueue to prioritize next day at 00:30 AM every day. It will prioritize only the hosts specified by tellusServers and cronServers.

Logs

Log location is configured by logFile in C:\Tellus\JobQueue\JobQueueConfig.properties:
`C:/Tellus/JobQueue/logs/JobQueue.log`

Restart

To restart JobQueue, simply restart TomEE Windows Server on tellusstgb01.