Java EE 7 アプリケーション設計ガイド

Batch Applications on Liberty 編

日本アイ・ビー・エム　システムズ・エンジニアリング株式会社

Disclaimer

- この資料は日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の正式なレビューを受けておりません。
- 当資料は、資料内で説明されている製品の仕様を保証するものではありません。
- 資料の内容には正確を期するよう注意しておりますが、この資料の内容は2015年11月現在の情報であり、製品の新しいリリース、PTFなどによって動作、仕様が変わる可能性があるのでご注意下さい。
- 今後国内で提供されるリリース情報は、対応する発表レターなどでご確認ください。
- ＩＢＭ、ＩＢＭロゴおよびibm.comは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれＩＢＭまたは各社の商標である場合があります。現時点でのＩＢＭの商標リストについては、www.ibm.com/legal/copytrade.shtmlをご覧ください。
- 当資料をコピー等で複製することは、日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の承諾なしではできません。
- 当資料に記載された製品名または会社名はそれぞれの各社の商標または登録商標です。
- JavaおよびすべてのJava関連の商標およびロゴは Oracleやその関連会社の米国およびその他の国における商標または登録商標です。
- Microsoft, Windows および Windowsロゴは、Microsoft Corporationの米国およびその他の国における商標です。
- Linuxは、Linus Torvaldsの米国およびその他の国における登録商標です。
- UNIXはThe Open Groupの米国およびその他の国における登録商標です。

目次

本文書の位置づけ

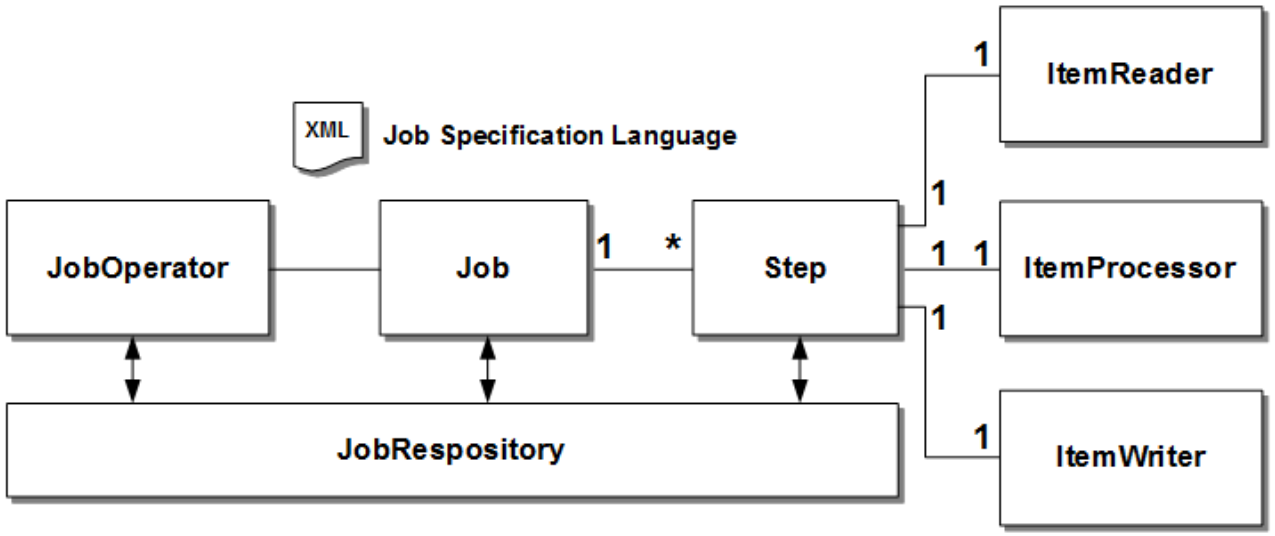
- この文書は、Java EE 7 アプリケーション設計ガイド シリーズにおいて、「JSR-352 Batch Applications for Java Platform」を対象に、Libertyでの実装にフォーカスした内容で記述したものです。
- 本文書は、WAS Libertyをランタイムとし、ジョブ管理インターフェイス、セキュリティやログ機能の実装、複数サーバーサポートまでをご紹介しています。
- 本文書で使用されるアプリケーションは、「JSR-352 Batch Applications for Java Platform編」で作成したサンプル・アプリケーションを使用しています。
- 今後、WAS Libertyでの実装例や役立つTipsなどをトピック毎にまとめ、developerWorksにて随時リリースする予定です。
  - http://www.ibm.com/developerworks/jp/websphere/category/was/

1. はじめに

1.1. JSR-352 Batch Applications for Java Platform
1.2. Libertyでの拡張機能

1.1. JSR-352 Batch Applications for Java Platform

- Java EE 7で標準化されたJavaにおけるバッチ処理の仕様
- 「ジョブ」と「ステップ」から構成
  - ステップは1つのItemReader、ItemProcessor、ItemWriterを保有
  - ジョブは、JobOperator によって起動
  - ジョブの実行状況は、JobRepositoryに格納

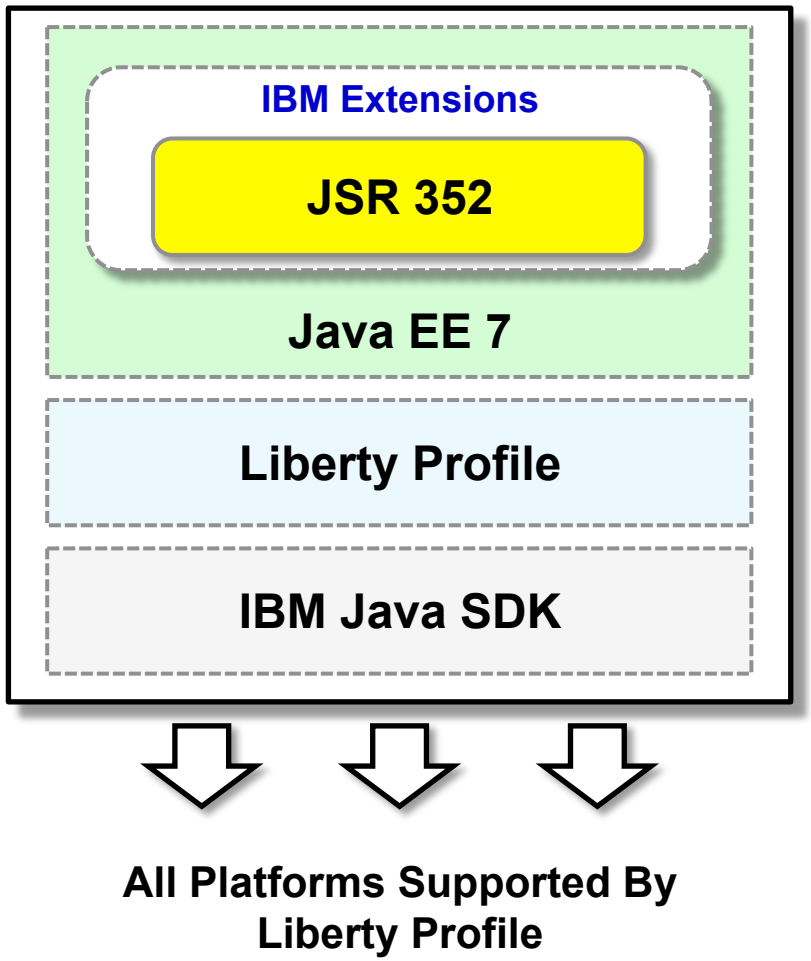


- ジョブはflow、split、decisionと呼ばれる実行要素を保有し、実行順序を制御
- ステップは複数データの逐次的処理を実行する「チャンク型」と、単体で完結する「バッチレット型」の2種類

1.2. Libertyでの拡張機能

- JSR-352 Batch Applications for Java Platformは Liberty Profile 8.5.5.6以降の環境で実行可能

- Liberty Profileでは、JSR-352の動作をサポートする以下の拡張機能を実装
  - JobOperatorに対するRESTインターフェイス
  - ジョブを操作するコマンド・ライン・クライアント
  - ロールベースのセキュリティ
  - ジョブログの実装
  - ディスパッチャーとエグゼキューターによる複数JVMサポート

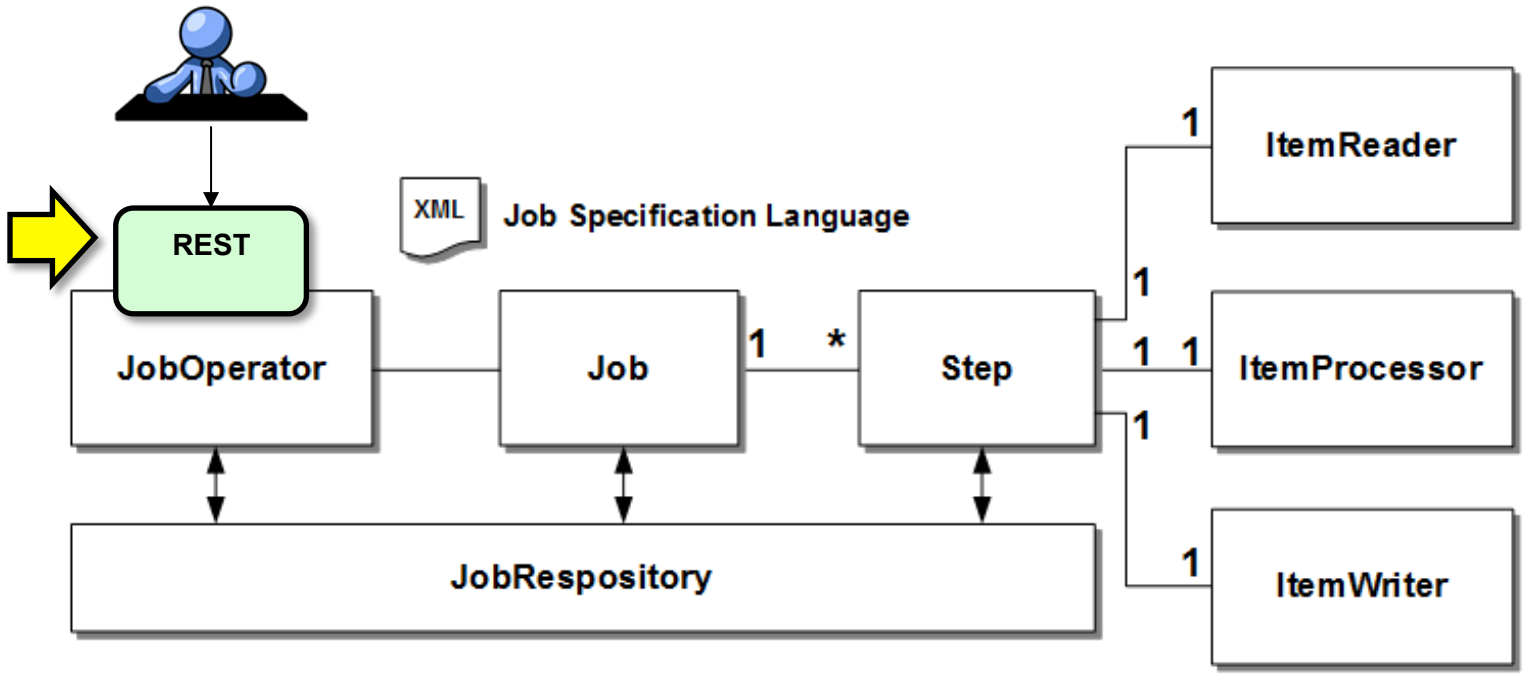


2. ジョブ管理のインターフェース

2.1. REST でのジョブ情報の取得・操作方法
2.2. batchManagerコマンドでの操作
2.3. JAX-RSクライアントの作成

2.1. REST でのジョブ情報の取得・操作方法

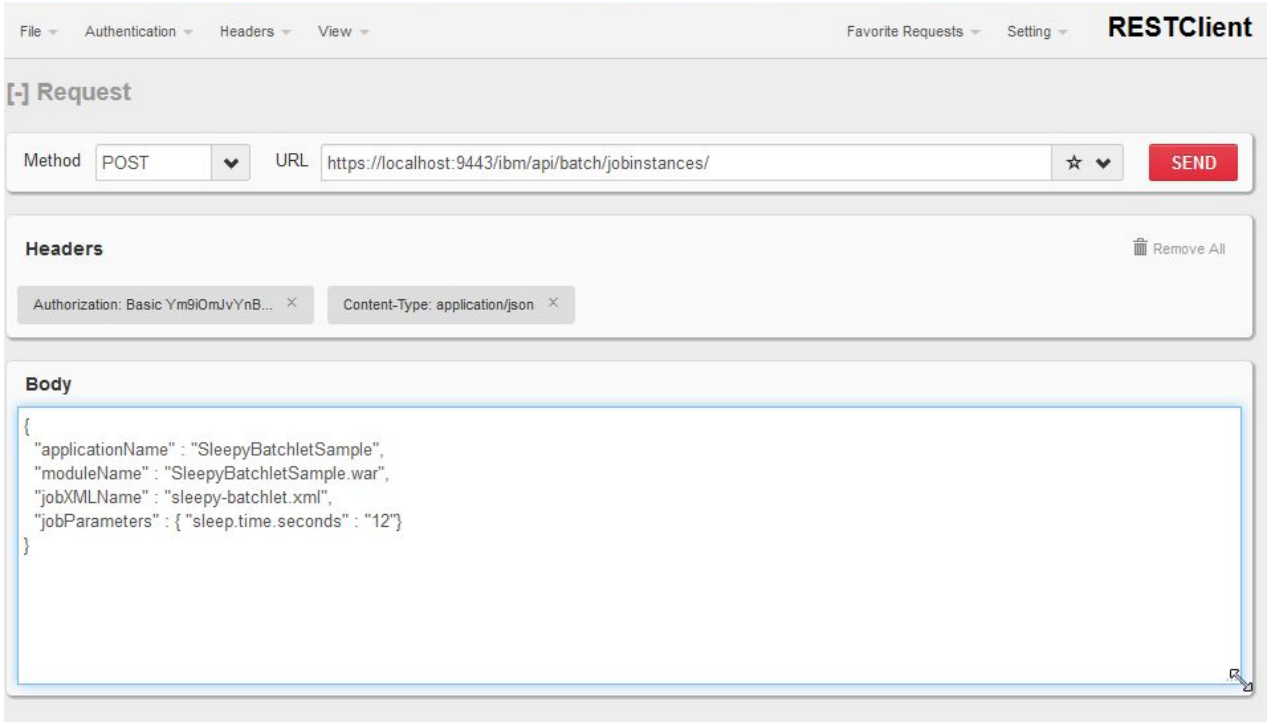
- JSR-352では、JobOperatorの呼び出し方については規定されていない。
- Liberty Profileは、バッチ・ジョブを管理するためのRESTfulインターフェースを提供
  - URLとコマンドでジョブの操作が可能
  - https://<host>:<port>/ibm/api/batch



2.1. REST でのジョブ情報の取得・操作方法

- RESTインターフェイスの使用例
  - ジョブのサブミット
    - Basic認証に必要な情報（例：bob/bobpwd）と、jsonでの情報受け渡しのために、以下のリクエスト・ヘッダを追加。
      - Authorization: Basic Ym9iOmJvYnB3ZA==
      - Content-Type: application/json
    - POSTのボディにアプリケーション名、モジュール名、JobXMLファイル、プロパティーなど、サブミットしたいジョブの情報を追加

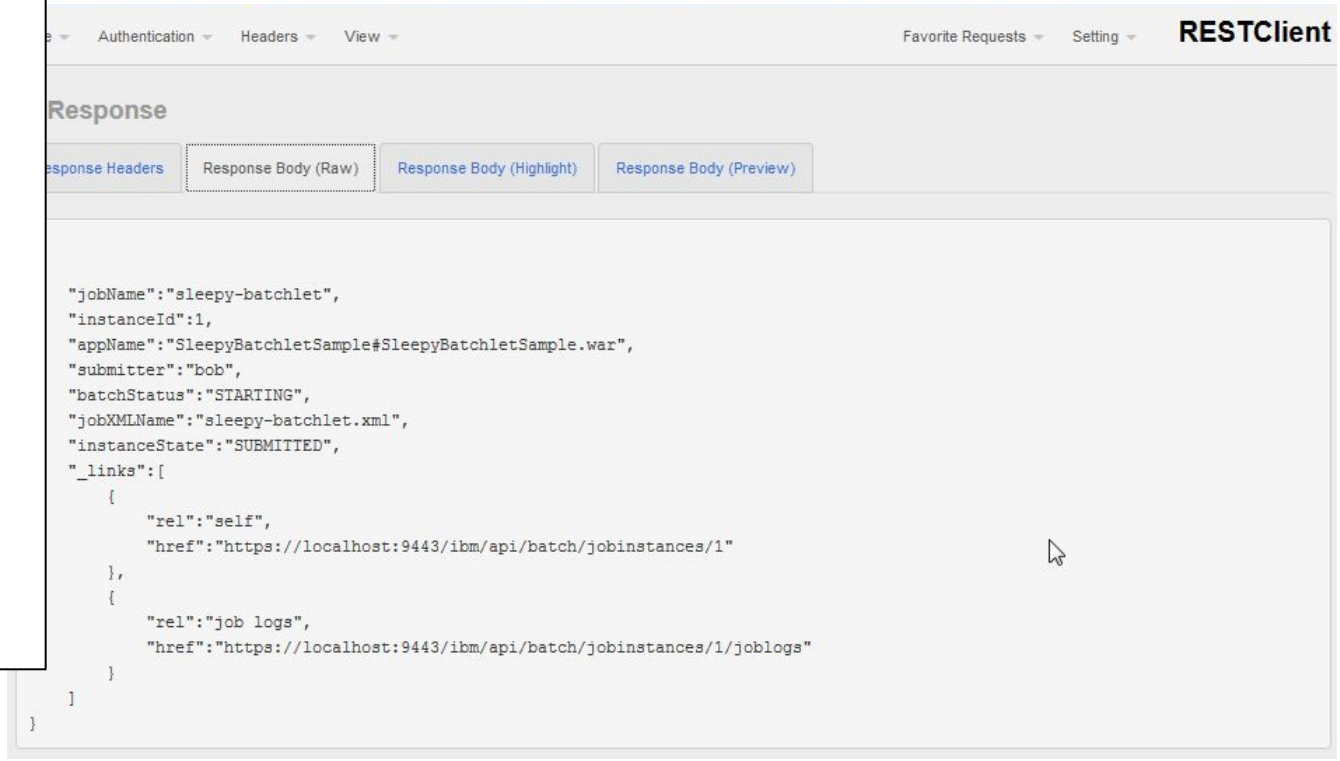
```
{
  "applicationName" : "SleepyBatchletSample",
  "moduleName" : "SleepyBatchletSample.war",
  "jobXMLName" : "sleepy-batchlet.xml",
  "jobParameters" : { "sleep.time.seconds" : "12"}
}
```



2.1. REST でのジョブ情報の取得・操作方法

– ジョブのサブミットを実行
  - POST https://<host>:<port>/ibm/api/batch/jobinstances/ を実行
  - サブミットしたジョブの情報がレスポンスとして返される


```
{
   "jobName":"sleepy-batchlet",
   "instanceId":1,
   "appName":"SleepyBatchletSample#SleepyBatchletSample.war",
   "submitter":"bob",
   "batchStatus":"STARTING",
   "jobXMLName":"sleepy-batchlet.xml",
   "instanceState":"SUBMITTED",
   "_links":[
      {
         "rel":"self",
         "href":"https://localhost:9443/ibm/api/batch/jobinstances/1"
      },
      {
         "rel":"job logs",
         "href":"https://localhost:9443/ibm/api/batch/jobinstances/1/joblogs"
      }
   ]
}
```

2.1. REST でのジョブ情報の取得・操作方法

- ジョブの停止

```
PUT /ibm/api/batch/jobexecutions/<JOB実行ID>?action=stop
```

- ジョブの再実行

```
PUT /ibm/api/batch/jobexecutions/<JOB実行ID>?action=restart
```

- ジョブの状況確認

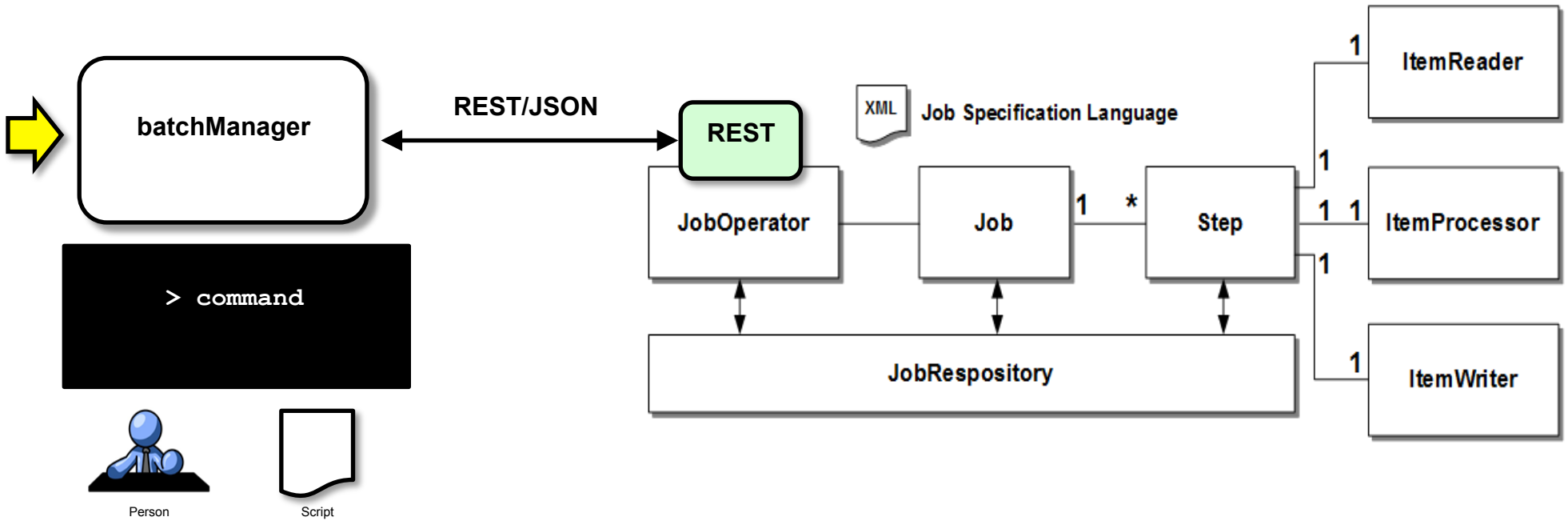
```
GET /ibm/api/batch/jobexecutions/<JOB実行ID>
```

- ジョブの実行結果取得

```
GET /ibm/api/batch/jobinstances/
```

2.2. batchManagerコマンドでの操作

- Libertyプロファイルで実行されるバッチ・ジョブを管理するためのコマンド・ライン・インターフェースが用意
  - –8.5.5.6以降のバージョンのLiberty　Profileの<WLPインストールディレクトリ>/bin 配下にコマンドが存在



2.2. batchManagerコマンドでの操作

- batchManagerコマンドの使用方法
  - batchManager {help|submit|stop|restart|status|getJobLog|listJobs|purge} [options]
    - help
      - 指定されたアクションのヘルプ情報を表示します。
    - submit
      - 新規バッチ・ジョブをサブミットします。
    - stop
      - バッチ・ジョブを停止します。
    - restart
      - バッチ・ジョブを再始動します。
    - status
      - ジョブの状況を表示します。
    - getJobLog
      - バッチ・ジョブのジョブ・ログをダウンロードします。
    - listJobs
      - ジョブ・インスタンスをリストします。
    - purge
      - ジョブ・インスタンスのすべてのレコードとログを消去するか、またはジョブ・インスタンス・レコードのリストを消去します。

2.2. batchManagerコマンドでの操作

- batchManagerコマンドの実行例
  - ジョブのサブミット

```
C:\NHT\liberty8558\wlp\bin>batchManager submit --user=bob --password=bobpwd --batchManager=localhost:9443 --trustSslCertificates --
jobXMLName=sleepy-batchlet.xml --applicationName=SleepyBatchletSample
[2016/05/13 17:04:22.493 +0900] CWWKY0101I: インスタンス ID 5 のジョブ sleepy-batchlet がサブミットされました。
[2016/05/13 17:04:22.493 +0900] CWWKY0106I: JobInstance:{"jobName":"sleepy-batchlet","instanceId":
5,"appName":"SleepyBatchletSample#SleepyBatchletSample.war","submitter":"bob","batchStatus":"STARTING","jobXMLName":"sleepy-
batchlet.xml","instanceState":"SUBMITTED"}
```

  - ジョブの停止

```
C:\NHT\liberty8558\wlp\bin>batchManager stop --user=bob --password=bobpwd --batchManager=localhost:9443 --trustSslCertificates --jobExecutionId=6
[2016/05/16 17:44:29.980 +0900] CWWKY0104I: インスタンス ID 6 のジョブ sleepy-batchlet に対して停止要求がサブミットされました。
```

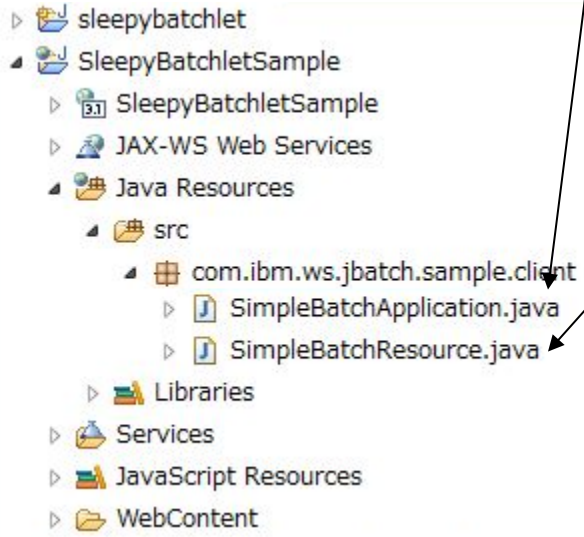
  - ジョブの状況確認

```
C:\NHT\liberty8558\wlp\bin>batchManager status --user=bob --password=bobpwd --batchManager=localhost:9443 --trustSslCertificates --jobExecutionId=7
[2016/05/16 17:50:55.179 +0900] CWWKY0106I: JobInstance:{"jobName":"sleepy-batchlet","instanceId":
7,"appName":"SleepyBatchletSample#SleepyBatchletSample.war","submitter":"jane","batchStatus":"STARTED","jobXMLName":"sleepy-
batchlet","instanceState":"DISPATCHED"}

[2016/05/16 17:50:55.194 +0900] CWWKY0107I: JobExecution:{"jobName":"sleepy-batchlet","executionId":7,"instanceId":
7,"batchStatus":"STARTED","exitStatus":"","createTime":"2016/05/16 17:50:45.457+0900","endTime":"","lastUpdatedTime":"2016/05/16 17:50:45.457
+0900","startTime":"2016/05/16 17:50:45.457 +0900","jobParameters":{"sleep.time.seconds":"60"},"restUrl":"https://localhost:9443/ibm/api/
batch","serverId":"localhost/C:/NHT/liberty8558/wlp/usr/defaultServer“,"logpath":"C:\\NHT\\liberty8558\\wlp\\usr\\servers\\defaultServer\\logs\\joblogs\\sleepy-
batchlet\\2016-05-16\\instance.7\\execution.7\\","stepExecutions":[{"stepExecutionId":
7,"stepName":"step1","batchStatus":"STARTED","exitStatus":"","stepExecution":"https://localhost:9443/ibm/api/batch/jobexecutions/7/stepexecutions/step1"}]}
```

2.3. JAX-RSクライアント作成

- JAX-RSのクライアントを自ら作成することも可能
  - POJOにアノテーションを付加するだけで RESTful Webサービスが作成可能
- Webモジュール側で必要なクラス
  - Applicationサブクラス
    - 作成するリソースクラスをRESTful Webサービスとして公開するためのクラス
    - 必要なのはApplicationクラスを継承したクラスと、ApplicationPathによるパスの指定のみ
  - RESTサービスクラス（リソースクラス）
    - RESTfulサービス本体を定義するクラス
    - Pathが定義されたクラスとJavaメソッド、アクセスするときに使うHTTPメソッド、メッセージボディの形式を指定




```
import javax.ws.rs.core.Application;
import javax.ws.rs.ApplicationPath;

@ApplicationPath("/rs")
public class SimpleBatchApplication extends Application{
    //実装はなし
}
```

```
(importは省略)
@Path("/jbatch")
public class SimpleBatchResource {
    @POST
    @Path("/start")
    @Produces("text/plain")
    public String startBatch(){
        JobOperator jo = BatchRuntime.getJobOperator();
        Properties property = new Properties();
        property.setProperty("sleep.time.seconds", "4");
        long jobExecId = jo.start("sleepy-batchlet", property);
        return "Batch has started. id = " + jobExecId;
    }
}
```

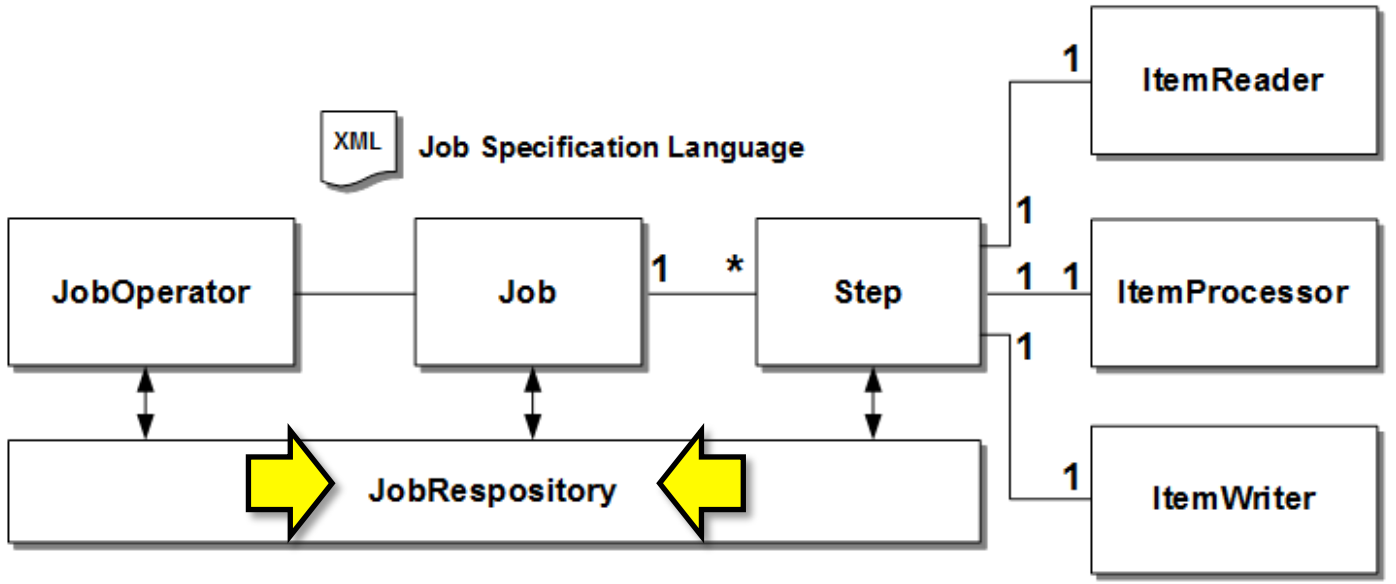
3. ジョブ・リポジトリの設定

3.1. ジョブ・リポジトリのパーシスト化

3.1. ジョブ･リポジトリのパーシスト化

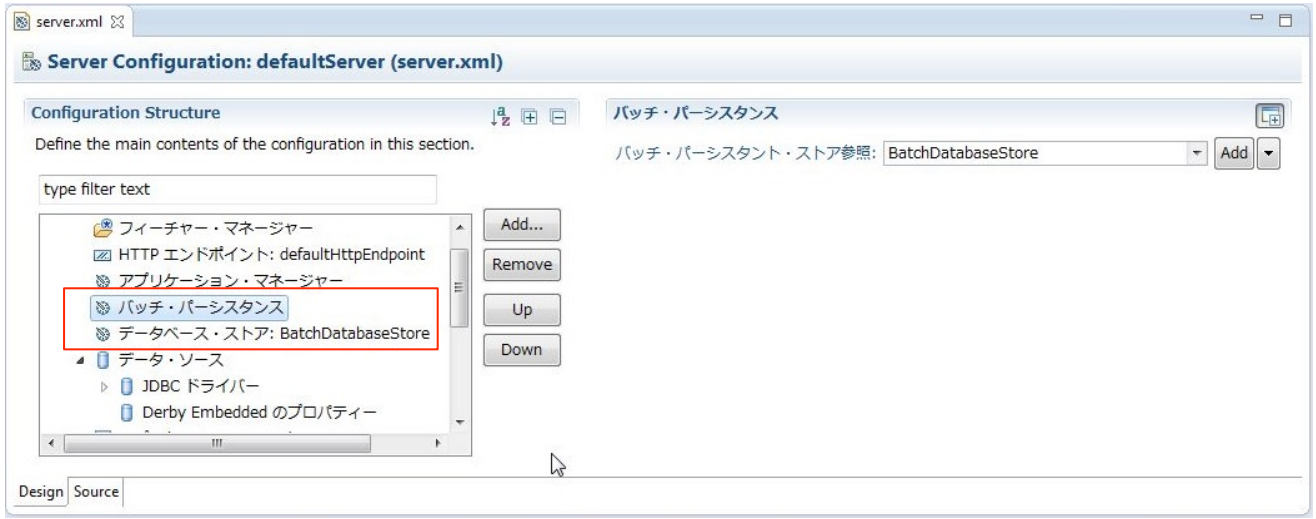
- ジョブ・リポジトリの選択肢
  1. インメモリ・ジョブリポジトリ
     ジョブのステータスを保持する必要がないケース、テスト環境向け
  2. ファイルベースのDerbyジョブリポジトリ
     ある程度は保持したいが、フルデータベース製品は必要ないケース
  3. RDB製品を用いたジョブリポジトリ
     パーシスタントを保ちたいケース

- パーシスタント・ストアの使用
  - 前に実行したジョブを
    停止しなければならないケースでも、
    再開されるジョブに
    該当するデータを提供して、
    ジョブ・インスタンスを
    再開させることが可能



3.1. ジョブ・リポジトリのパーシスト化

- Derbyジョブリポジトリのパーシスト化の例
  - server.xmlに<batchPersistence>、<databaseStore>の情報を追加



server.xml

```
<!-- The default self-signed SSL certificate in this example is intended only for development use and not for production.-->
<batchPersistence jobStoreRef="BatchDatabaseStore"></batchPersistence>
<databaseStore id='BatchDatabaseStore' createTables="false" dataSourceRef="batchdb"></databaseStore>
<dataSource id="batchdb">
  <jdbcDriver>
     <library>
          <fileset dir="${server.config.dir}/resources/derby" includes="derby.jar" />
      </library>
  </jdbcDriver>
  <properties.derby.embedded createDatabase="create" databaseName="${server.config.dir}/resources/BATCHDB"
       password="pass" user="user" />
</dataSource>
```

3.1. ジョブ・リポジトリのパーシスト化

- ジョブを実行

Markers | Properties | Servers | Data Source Explorer | Snippets | Console | Annotations | Java EE Batch Job Logs

Job	Instance	Execution	Exit status	End time	Server
sleepy-batchlet	1	1	COMPLETED	2016/05/27 13:58:08.447 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost
sleepy-batchlet	2	2	COMPLETED	2016/05/27 13:58:23.283 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost
sleepy-batchlet	3	3	COMPLETED	2016/05/27 13:58:36.231 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost

JVM再起動

パーシスタント設定がなければ
再起動によりジョブIDの情報は削除されるため、
同じジョブを実行しても
ジョブID=0からスタート

- 再起動後に同じジョブを再び実行しても、ジョブIDが0に戻ることなく、再起動前のジョブIDから再開

Markers | Properties | Servers | Data Source Explorer | Snippets | Console | Annotations | Java EE Batch Job Logs

Job	Instance	Execution	Exit status	End time	Server
sleepy-batchlet	1	1	COMPLETED	2016/05/27 13:58:08.447 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost
sleepy-batchlet	2	2	COMPLETED	2016/05/27 13:58:23.283 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost
sleepy-batchlet	3	3	COMPLETED	2016/05/27 13:58:36.231 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost
sleepy-batchlet	4	4	COMPLETED	2016/05/27 14:02:32.290 +0...	WebSphere Application Server V8.5.5.8 Liberty Profile at localhost

4. セキュリティ

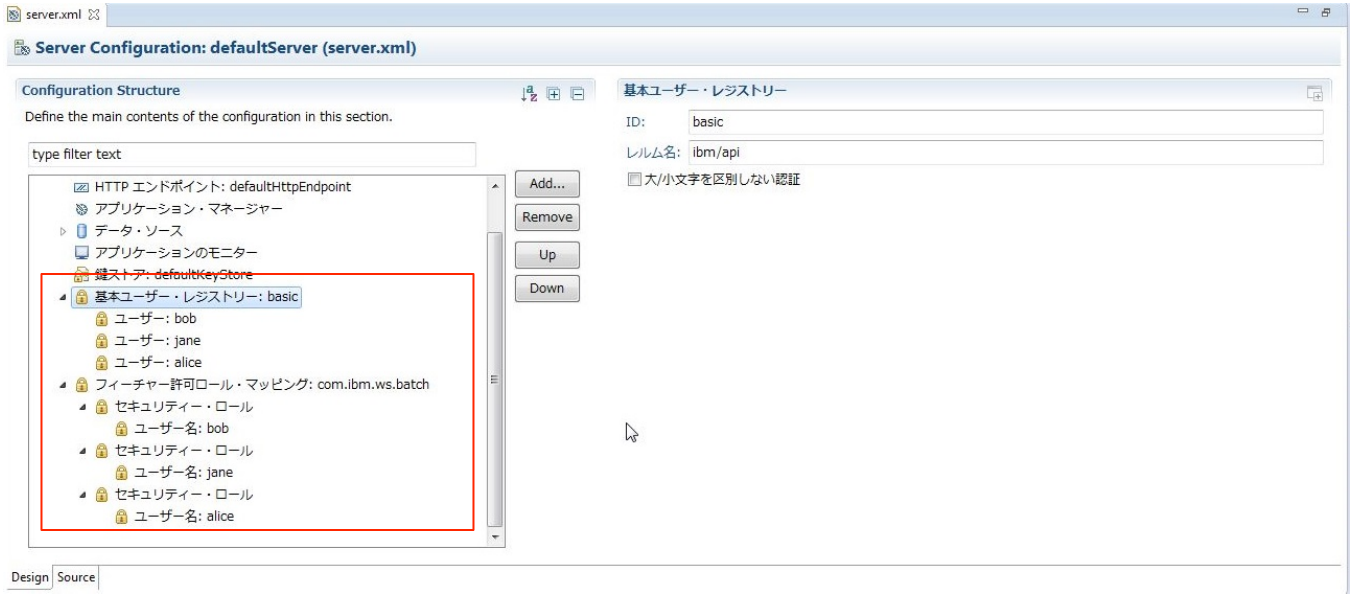
4.1. ロールによる権限
4.2. ロールの設定

4.1. ロールによる権限

- Liberty Profileはバッチ管理操作に対してロール・ベースの権限付与が可能
  - batchAdmin
    - **すべてのバッチ操作**の権限
    - 自身がサブミットしたジョブだけでなく、任意のユーザーのジョブの停止と再開、 稼働状況やログの表示、パージが可能
  - batchSubmitter
    - **新規ジョブのサブミット**を行う権限
    - 自分のジョブについてのみ、停止、再開、パージなどのバッチ操作を実行。他のユーザーのジョブの表示および変更は不可
  - batchMonitor
    - すべてのジョブに関する**読み取り専用**権限
    - 自分でジョブをサブミットすることはできず、ジョブの停止、再開、パージも不可

4.2. ロールの設定

- ロールの設定
  - 基本ユーザー・レジストリーの設定
    - レルム名は「ibm/api」
    - ユーザーとパスワードを指定
    - パスワードは暗号化も可能
  - フィーチャー許可ロール・マッピングの設定
    - 各セキュリティ・ロールを指定し、紐付けるユーザー名を指定



server.xml

```
<basicRegistry id="basic" realm="ibm/api">
        <user name="bob" password="bobpwd"/>
        <user name="jane" password="janepwd"/>
        <user name="alice" password="alicepwd"/>
</basicRegistry>
<authorization-roles id="com.ibm.ws.batch">
        <security-role name="batchAdmin" ><user name="bob"/></security-role>
        <security-role name="batchSubmitter" ><user name="jane"/></security-role>
        <security-role name="batchMonitor" ><user name="alice"/></security-role>
</authorization-roles>
```

4.2. ロールの設定

- ロールを利用したジョブ実行制御の例
  - batchAdminであるBobが実行したジョブを、batchSubmitterのJaneが停止しようとすると、ステータスコード401((Unauthorized)が返る

```
C:\NHT\liberty8558\wlp\bin>batchManager stop --user=jane --password=janepwd --batchManager=localhost:9443 --trustSslCertificates
--jobExecutionId=1

[2016/05/25 14:15:21.222 +0900] エラー:  Server returned HTTP response code: 401 for URL: https://localhost:9443/ibm/api/batch/
jobexecutions/1?action=stop: [Error 401: CWWKY0302W: ユーザー jane には、ジョブ・インスタンス 1 に関連付けられたバッチ操作を実行
する許可が与えられていません。]
```

  - batchAdminのBobはジョブを停止可能

```
C:\NHT\liberty8558\wlp\bin>batchManager stop --user=bob --password=bobpwd --batchManager=localhost:9443 --trustSslCertificates
--jobExecutionId=0
[2016/05/25 14:14:29.973 +0900] CWWKY0104I: インスタンス ID 0 のジョブ sleepy-batchlet に対して停止要求がサブミットされました。
```

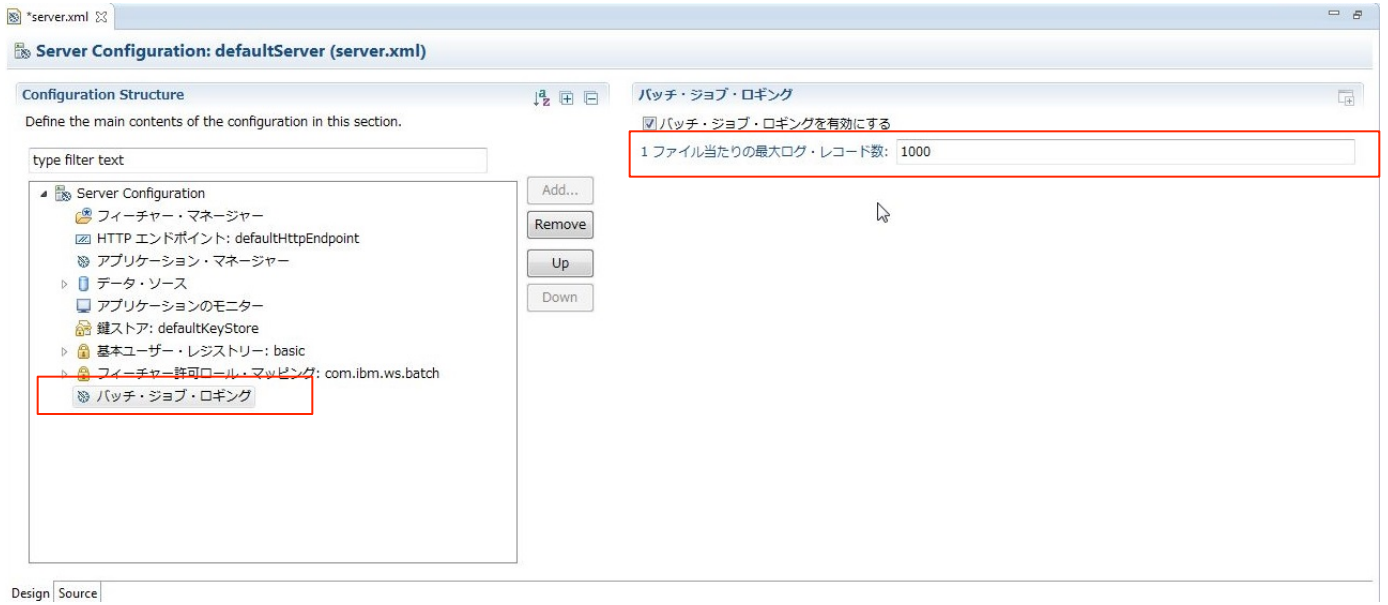
5. ジョブログ

5.1. ジョブログのローテーション
5.2. ログ出力レベルの変更

5.1. ジョブログのローテーション

- ジョブログの出力先
  - <Liberty serverディレクトリ>/logs/joblogs/<ジョブ名>/日付（YYYY-MM-DD）/instance.<インスタンスID>/executon.<実行ID>/part.<シーケンス番号>.log
  - パーシスタントの設定がない場合、同じ日にサーバーを再起動し、インスタンスIDや実行IDが0に戻った状態で同じジョブを実行するとログは上書き

- ジョブログのローテーション
  - デフォルトではログに1000行記載されることにローテーション
  - バッチ・ジョブ・ロギングのmaxRecordsの設定でローテーション対象となる記載行数を変更可能

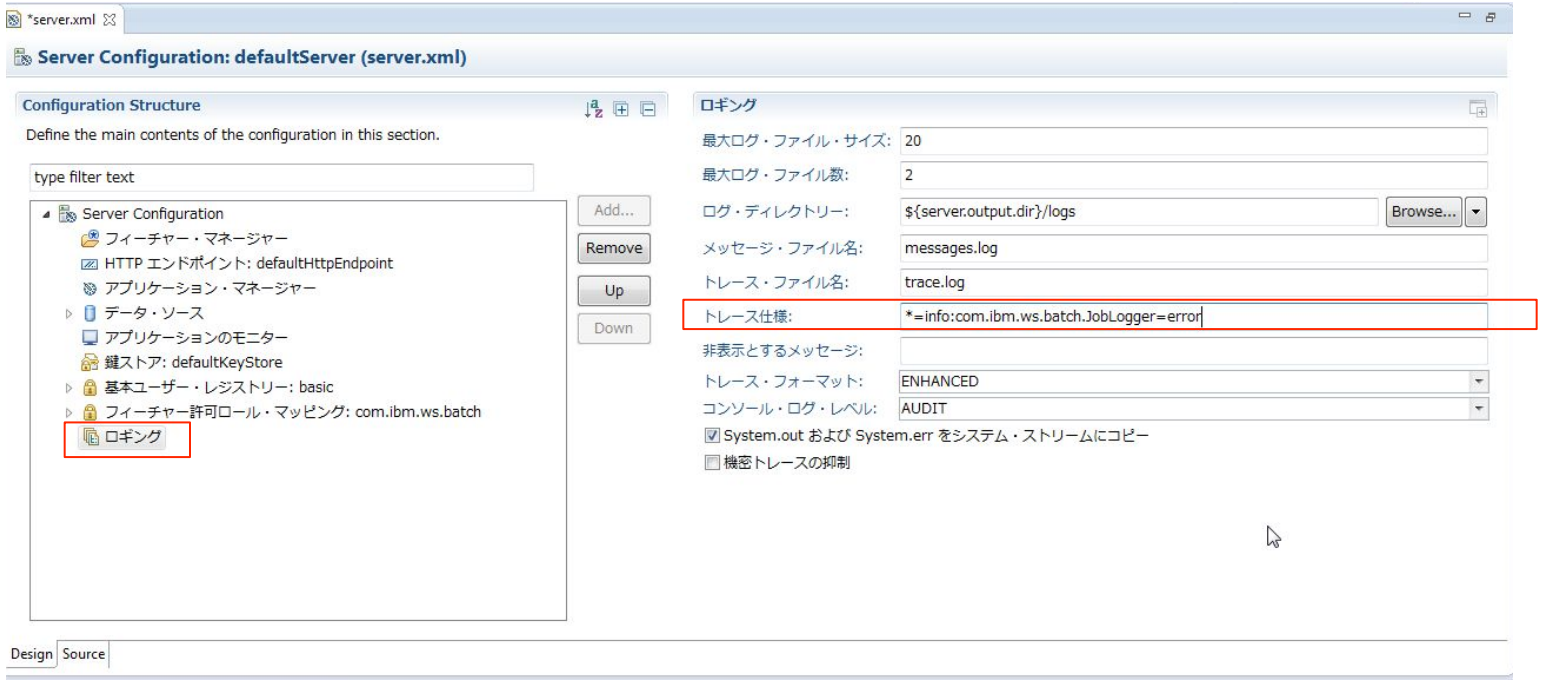


server.xml

```
<batchJobLogging enabled="true" maxRecords="5"></batchJobLogging>
```

5.2. ログ出力レベルの変更

- ジョブログの出力レベルの変更
  - ジョブログのロガーは、デフォルトでLevel.FINE
  - ジョブログのロガーの出力レベルは、サーバーのトレース仕様の変更により設定可能
    - ロギングの「トレース仕様」の欄より com.ibm.ws.batch.JobLoggerの設定を変更



server.xml

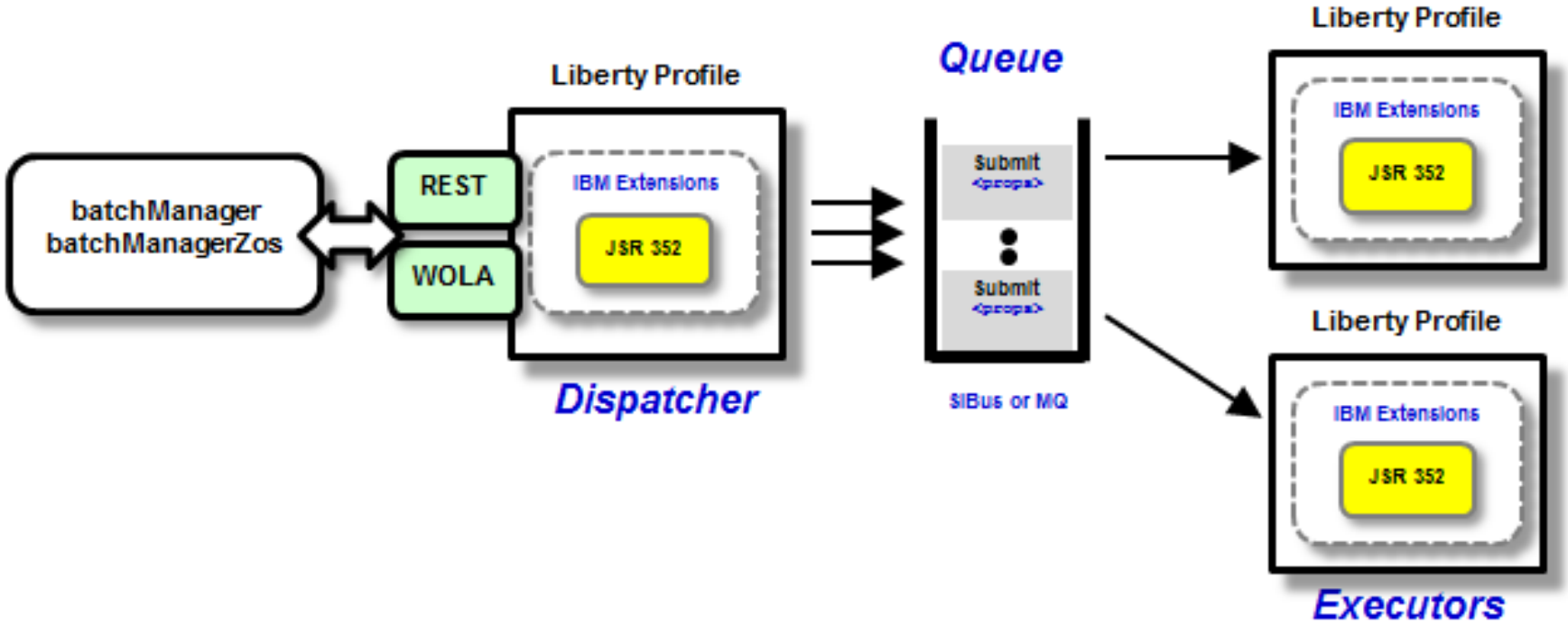
```
<logging traceSpecification='*=info:com.ibm.ws.batch.JobLogger=error'></logging>
```

6. 複数サーバー・サポート

6.1. ディスパッチャーとエグゼキューター
6.2. バッチJMSメッセージング・エンジン・サーバーの構築
6.3. バッチ・ディスパッチャーの構築
6.4. バッチ・エグゼキューターの構築
6.5. 稼動確認

6.1. ディスパッチャーとエグゼキューター

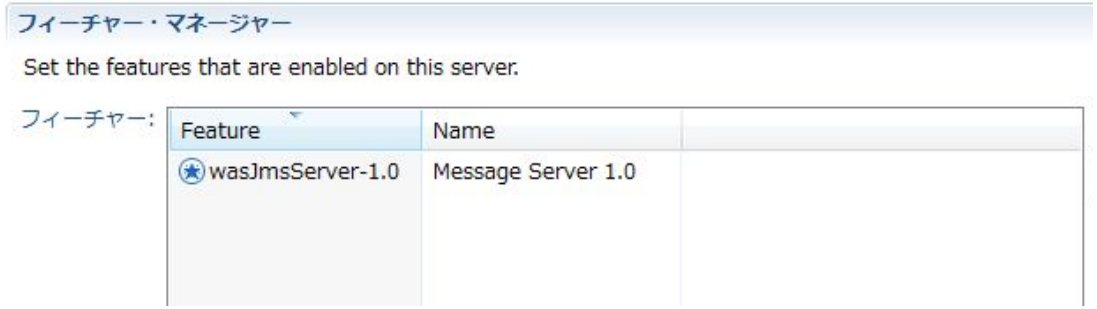
- バッチ環境の複数サーバー・サポート
  - 負荷を分散することで、大量のバッチ処理を行うことが可能
- リクエストを受け付けるバッチ・ディスパッチャーと、処理を実行するバッチ・エグゼキューターに役割分担
- バッチ・ディスパッチャーとバッチ・エグゼキューターはJava Messaging Service (JMS) を使用して通信
- 以下は、ディスパッチャーを1つ、エグゼキューターを2つで構成した例
  - 要件に応じて柔軟な構成をとることが可能



6.2. バッチJMSメッセージング・エンジン・サーバーの構築

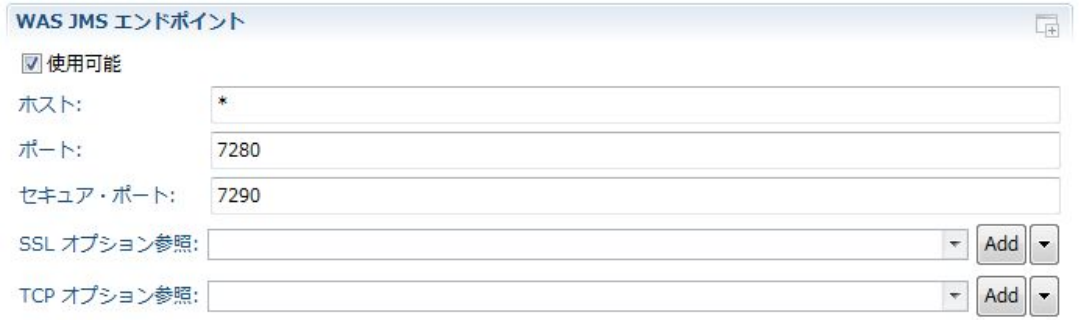
- wasJmsServer-1.0 フィーチャーをserver.xmlに追加

```
<featureManager>
  <feature>wasJmsServer-1.0</feature>
</featureManager>
```



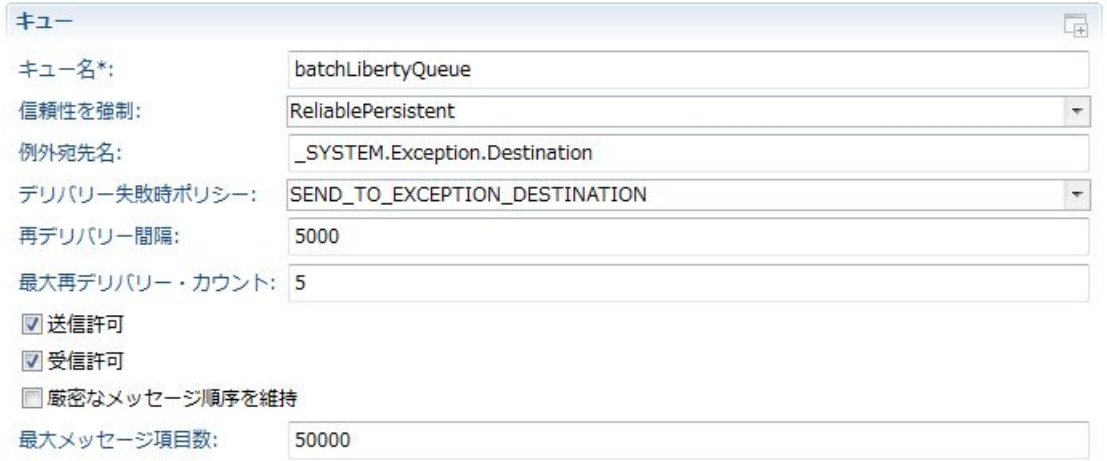
- JMSクライアントがアクセスするための JMSエンドポイントを定義

```
<wasJmsEndpoint  enabled="true"  host="*"  wasJmsPort="7280"
    wasJmsSSLPort="7290"/>
```



- バッチ・ディスパッチャーやバッチ・エグゼキューターとの 通信に使われるキューを構成

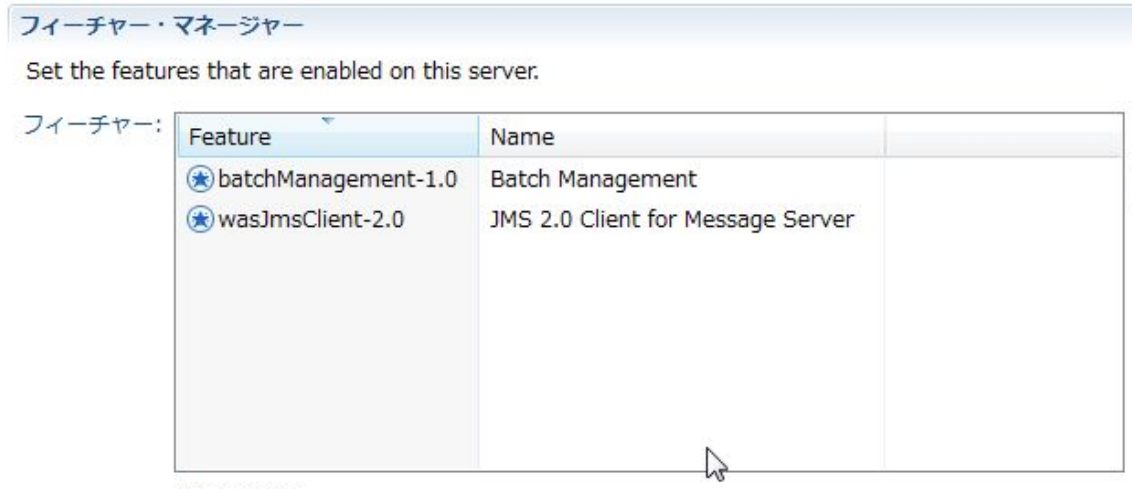
```
<messagingEngine>
  <queue  forceReliability="ReliablePersistent"  id="batchLibertyQueue"
      receiveAllowed="true"  sendAllowed="true"/>
</messagingEngine>
```



6.3. バッチ・ディスパッチャーの構築

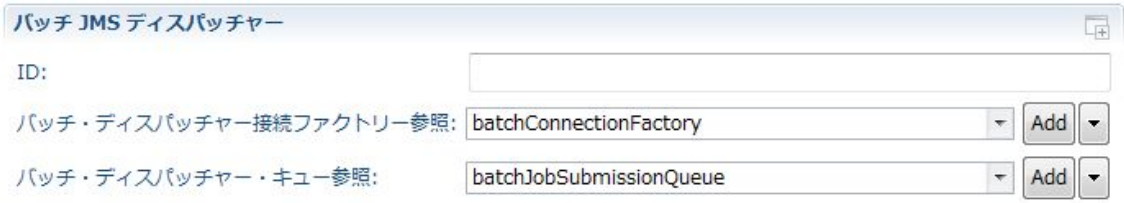
- batchManager-1.0とwasJmsClient-2.0フィーチャーを server.xmlに追加

```
<featureManager>
   <feature>wasJmsClient-2.0</feature>
   <feature>batchManagement-1.0</feature>
</featureManager>
```



- バッチ・ディスパッチャーとして構築するために batchJmsDispatcherエレメントを追加

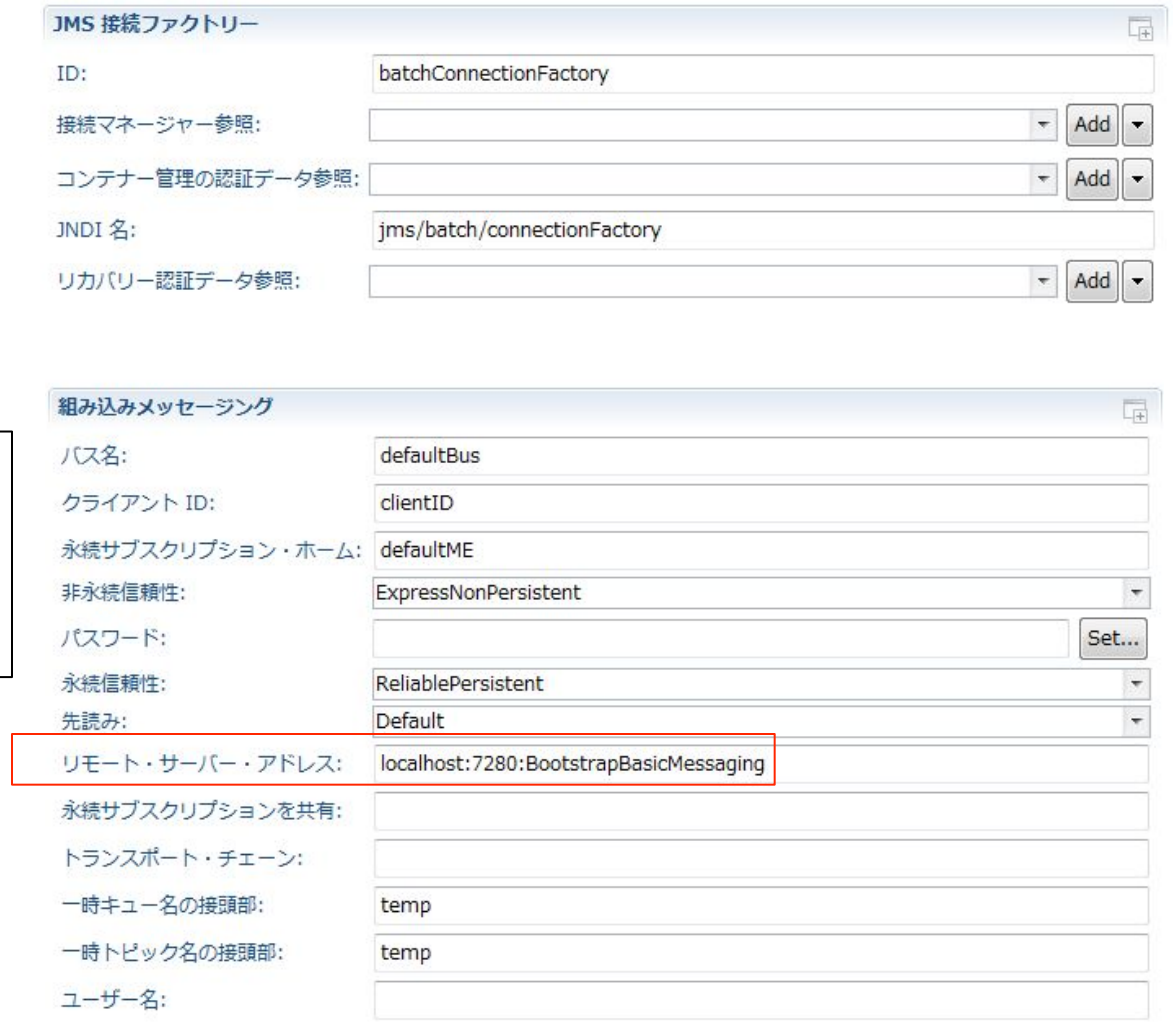
```
<batchJmsDispatcher
connectionFactoryRef="batchConnectionFactory"
queueRef="batchJobSubmissionQueue"/>
```



6.3. バッチ・ディスパッチャーの構築

- JMS接続ファクトリーの設定を追加
  - remoteServerAddress エレメントには、バッチJMSメッセージング・エンジン・サーバーの構築で定義したポート番号を指定

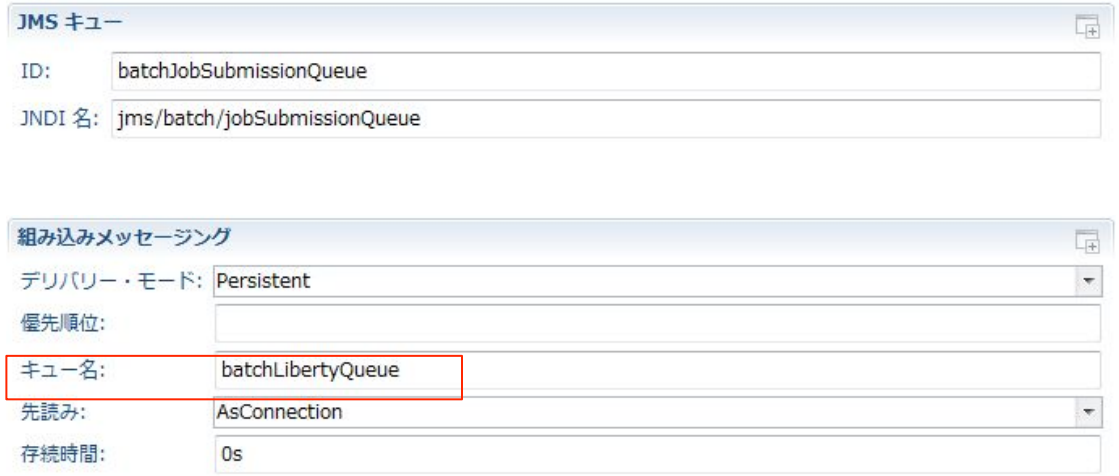
```
<jmsConnectionFactory id="batchConnectionFactory"
jndiName="jms/batch/connectionFactory">
<properties.wasJms remoteServerAddress="localhost:
7280:BootstrapBasicMessaging"/>
</jmsConnectionFactory>
```



6.3. バッチ・ディスパッチャーの構築

- JMSキューの設定を追加
  - queueNameエレメントは
    バッチJMSメッセージング・エンジン・サーバーの
    構築で定義したqueueのidを指定

```
<jmsQueue id="batchJobSubmissionQueue" jndiName="jms/
batch/jobSubmissionQueue">
<properties.wasJms deliveryMode="Persistent"
queueName="batchLibertyQueue"/>
</jmsQueue>
```



6.3. バッチ・ディスパッチャーの構築

- httpEndpointエレメントでHTTP ポートを構成
  - RESTでジョブをサブミットする際、URLでこのポートを指定

```
<httpEndpoint httpPort="9082" httpsPort="9445"
id="defaultHttpEndpoint"/>
```

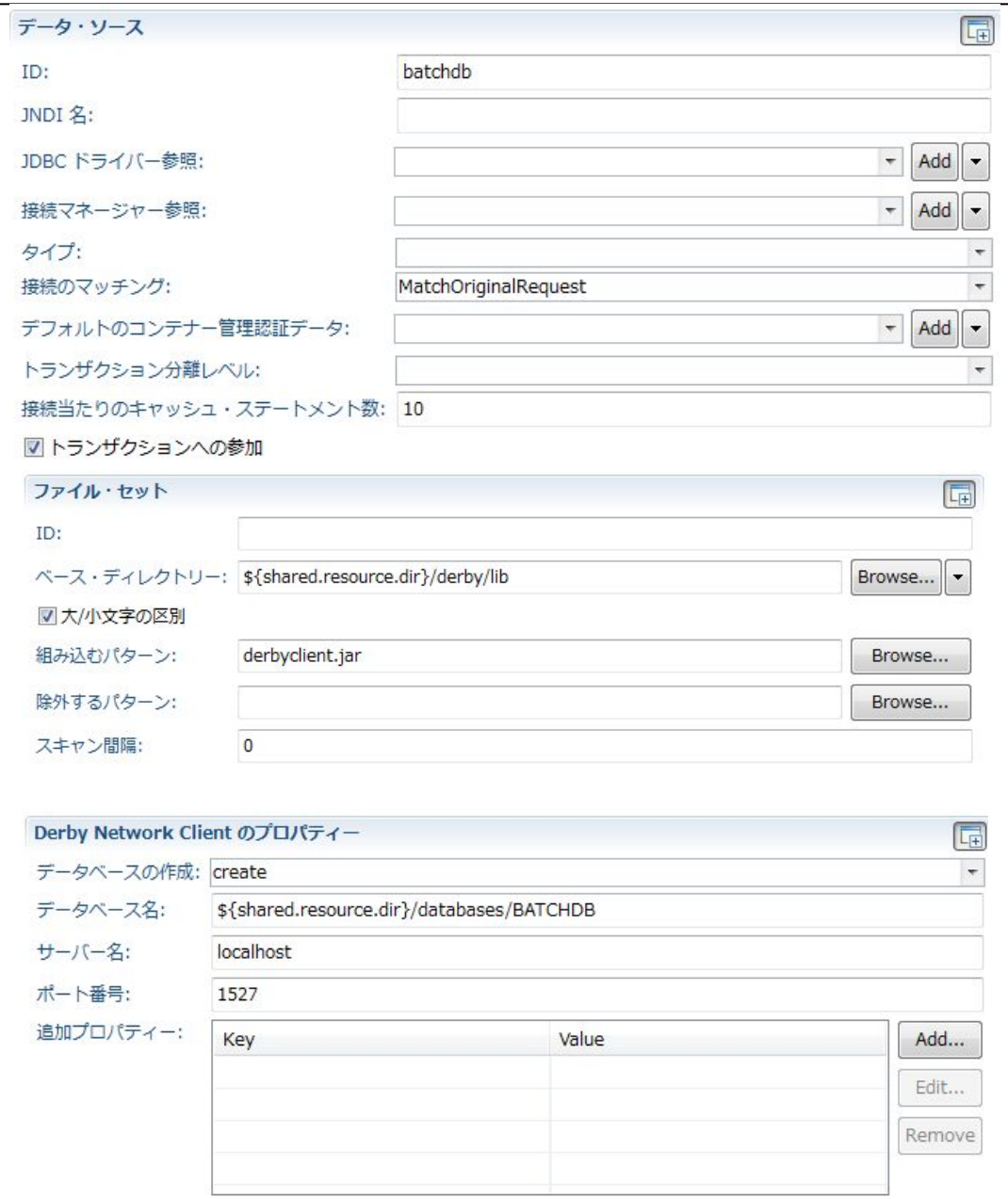
- バッチ・パーシスタンスを構成
  - batchPersistenceとdatabaseStoreを定義
  - p. 19 参照

```
<batchPersistence jobStoreRef="BatchDatabaseStore"></
batchPersistence>
<databaseStore id="BatchDatabaseStore" createTables="true"
dataSourceRef="batchdb"></databaseStore>
```

6.3. バッチ・ディスパッチャーの構築

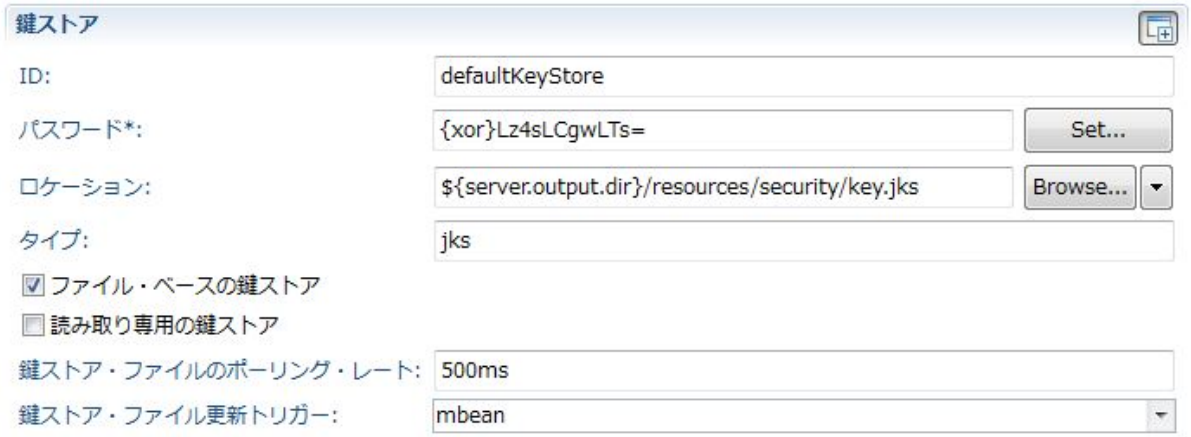
- ジョブ・リポジトリのためのDBを構築
  - JDBC ドライバーの共有ライブラリとデータソースを構成
  - 「Java EE 7 アプリケーション設計ガイド(JSR-352 Batch Applications for Java Platform 編)のp. 65を参照

```
<dataSource id="batchdb">
   <jdbcDriver>
     <library>
        <fileset dir="${shared.resource.dir}/derby/lib"
includes="derbyclient.jar" />
      </library>
   </jdbcDriver>
  <properties.derby.client createDatabase="create"
    databaseName="${shared.resource.dir}/databases/BATCHDB"
      serverName="localhost" portNumber="1527"
        password="pass" user="user" />
</dataSource>
```



6.3. バッチ・ディスパッチャーの構築

- キーストアとユーザーの設定
  - KeystoreとbasicRegistryの構築
  - 「Java EE 7 アプリケーション設計ガイド (JSR-352 Batch Applications for Java Platform 編)のp. 79を参照




```
<keyStore id="defaultKeyStore" password="{xor}Lz4sLCgwLTs=" />
<basicRegistry id="basic" realm="ibm/api">
  <user name="bob" password="bobpwd" />
</basicRegistry>
```

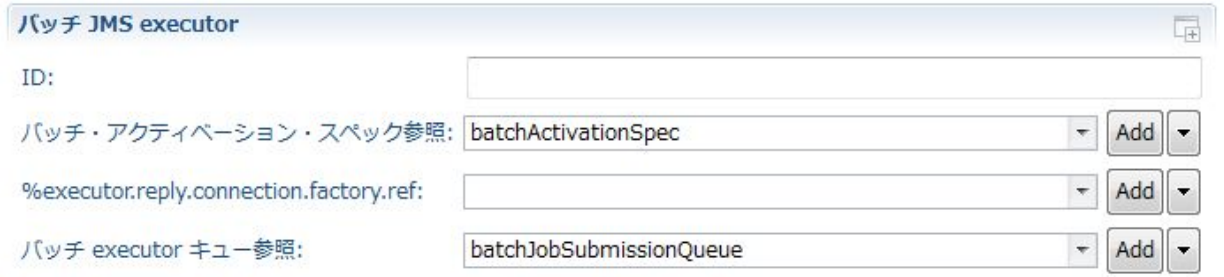
- フィーチャー許可ロール・マッピングの設定
  - 各セキュリティ・ロールを指定し、紐付けるユーザー名を指定
  - p.23 参照

```
<authorization-roles id="com.ibm.ws.batch">
  <security-role name="batchAdmin">
    <user name="bob" />
  </security-role>
</authorization-roles>
```

6.4. バッチ・エグゼキューターの構築

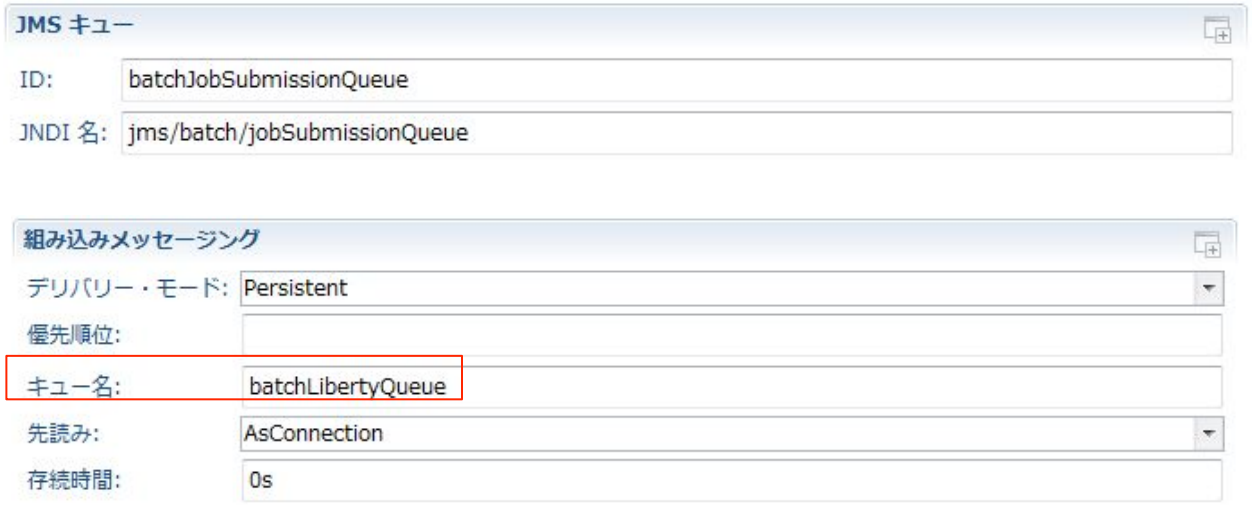
- バッチ・ディスパッチャーと異なるのはJMS接続の設定のみ
- バッチ・エグゼキューターとして構築するためにbatchJmsExecutorエレメントを追加

```
<batchJmsExecutor activationSpecRef="batchActivationSpec"
queueRef="batchJobSubmissionQueue"/>
```



- JMSキューの設定を追加
  - queueNameエレメントは
    バッチJMSメッセージング・エンジン・サーバーの
    構築で定義したqueueのidを指定

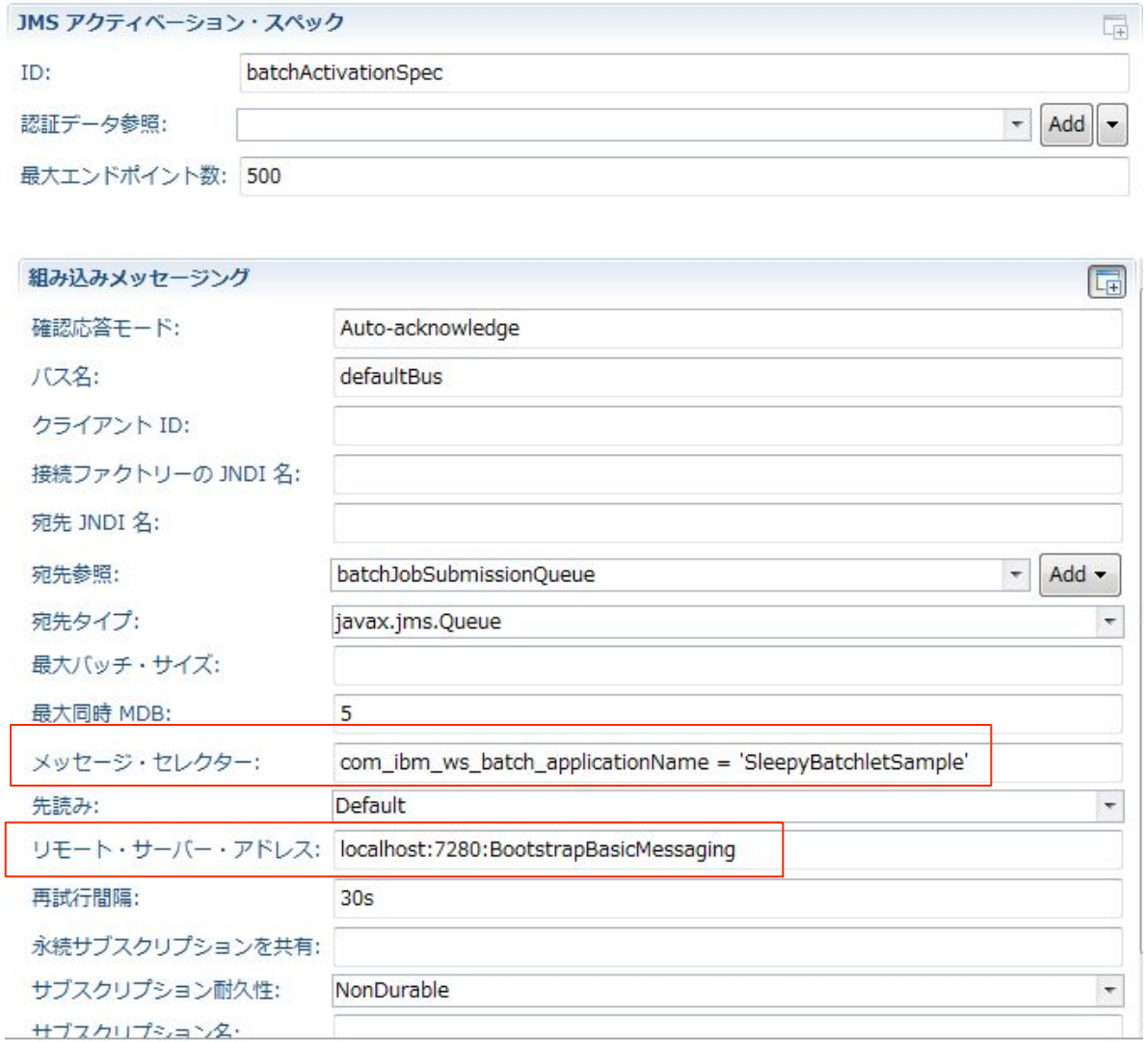
```
<jmsQueue id="batchJobSubmissionQueue" jndiName="jms/
batch/jobSubmissionQueue">
<properties.wasJms deliveryMode="Persistent"
queueName="batchLibertyQueue"/>
</jmsQueue>
```



6.4. バッチ・エグゼキューターの構築

- アクティベーション・スペックの設定を追加
  - messageSelector エレメントで
    バッチ・エグゼキューターで処理されるジョブの
    フィルタリングを実施
  - 例は「com_ibm_ws_batch_applicationName」
    属性を指定し、SleepyBatchletSampleアプリを
    実行するように設定
  - remoteServerAddress エレメントには、
    バッチJMSメッセージング・エンジン・サーバーの
    構築で定義したポート番号を指定

```
<jmsActivationSpec id="batchActivationSpec">
<properties.wasJms destinationRef="batchJobSubmissionQueue"
  remoteServerAddress="localhost:7280:BootstrapBasicMessaging"
    messageSelector="com_ibm_ws_batch_applicationName =
      'SleepyBatchletSample'"/>
</jmsActivationSpec>
```



6.5. 稼動確認

- アプリケーションのデプロイ
  - デプロイ先はエグゼキューターのサーバー

- Derby Network Server を起動
  - <Derbyインストールディレクトリ>/bin/startNetworkServer　コマンドを実行

```
C:\NHT\ServerSample\wlp2\usr\shared\resources\derby\bin>startNetworkServer
Fri Jun 17 17:46:45 JST 2016 : セキュリティ・マネージャがBasicサーバー・セキュリティ・ポリシーを使用してインストールされました。
Fri Jun 17 17:46:46 JST 2016 : Apache Derby Network Server - 10.12.1.1 - (1704137)が起動し、ポート1527で接続の受入れ準備が完了しました。
```

- バッチをサブミット
  - サブミット先はディスパッチャーのHTTPポート

```
C:\NHT\ServerSample\wlp2\bin>batchManager submit --user=bob --password=bobpwd --batchManager=localhost:9445 --trustSslCertificates --
jobXMLName=sleepy-batchlet.xml --applicationName=SleepyBatchletSample

[2016/06/17 17:56:44.573 +0900] CWWKY0101I: インスタンス ID 301 のジョブ (NOT SET) がサブミットされました。
[2016/06/17 17:56:44.573 +0900] CWWKY0106I: JobInstance:{"jobName":"(NOT SET)","instanceId":
301,"appName":"SleepyBatchletSample#SleepyBatchletSample.war","submitter":"bob","batchStatus":"STARTING","jobXMLName":"sleepy-
batchlet.xml","instanceState":"SUBMITTED"}
```

END