

Estudo de caso Gate2all sob a arquitetura JEE



CDI

JMS 2.0



EJB 3

Maior controle no uso do poll de threads

WildFly 9.0.1.Final | Messages: 0 | admin | [Logout](#)

Configuration: Subsystems » Subsystem: EE

[Close](#)

EE SUBSYSTEM SERVICES default

Context Service Executor Attributes Need Help?

Scheduled Executor

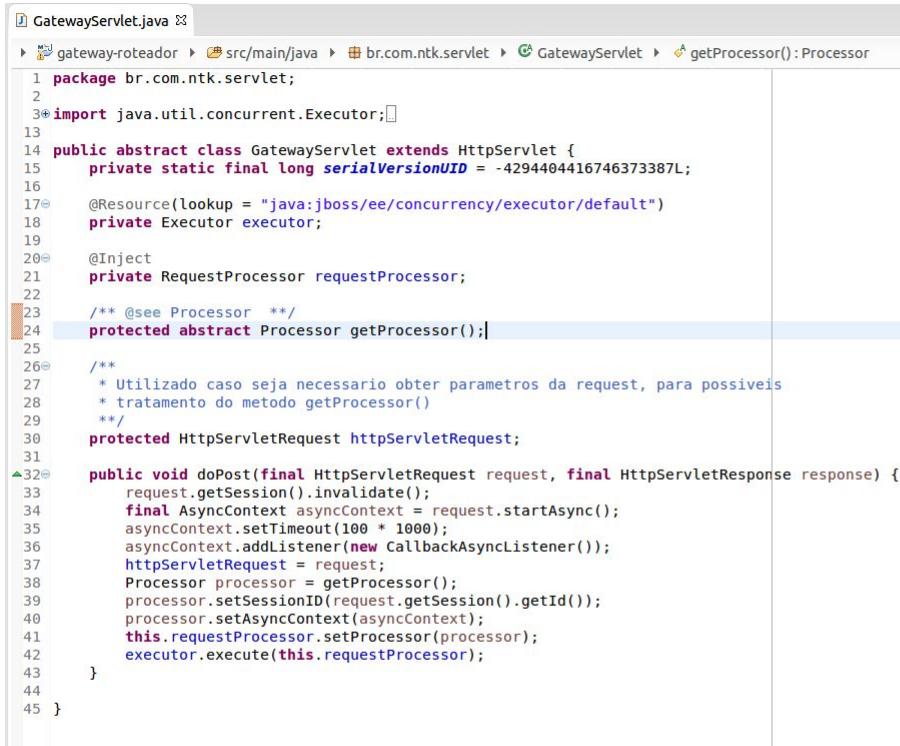
Thread Factories

Edit

Context service:	default
Core threads:	5
Hung task threshold:	60000
Jndi name:	java:jboss/ee/concurrency/executor/default
Keepalive time:	5000
Long running tasks:	false
Max threads:	25
Queue length:	0
Reject policy:	ABORT
Thread factory:	

« < 1-1 of 1 > »

Modelo de uso com JEE



A screenshot of a Java code editor showing the file `GatewayServlet.java`. The code is an abstract class that extends `HttpServlet`. It includes annotations for `@Resource` and `@Inject`, and a protected abstract method `getProcessor()`. The code also contains comments and logic for handling `HttpServletRequest` and `HttpServletResponse` using an `Executor`.

```
1 package br.com.ntk.servlet;
2
3 import java.util.concurrent.Executor;
4
5 public abstract class GatewayServlet extends HttpServlet {
6     private static final long serialVersionUID = -4294404416746373387L;
7
8     @Resource(lookup = "java:jboss/ee/concurrency/executor/default")
9     private Executor executor;
10
11     @Inject
12     private RequestProcessor requestProcessor;
13
14     /** @see Processor */
15     protected abstract Processor getProcessor();
16
17     /**
18      * Utilizado caso seja necessario obter parametros da request, para possiveis
19      * tratamento do metodo getProcessor()
20      */
21     protected HttpServletRequest httpServletRequest;
22
23     public void doPost(final HttpServletRequest request, final HttpServletResponse response) {
24         request.getSession().invalidate();
25         final AsyncContext asyncContext = request.startAsync();
26         asyncContext.setTimeout(100 * 1000);
27         asyncContext.addListener(new CallbackAsyncListener());
28         httpServletRequest = request;
29         Processor processor = getProcessor();
30         processor.setSessionID(request.getSession().getId());
31         processor.setAsyncContext(asyncContext);
32         this.requestProcessor.setProcessor(processor);
33         executor.execute(this.requestProcessor);
34     }
35 }
```

Modelo de uso atual

```
18  @WebListener
19  public class InitializeServletListener implements ServletContextListener {
20
21      public void contextInitialized(ServletContextEvent sce) {
22
23          Executor executor = new ThreadPoolExecutor(20, 50, 5, TimeUnit.MINUTES,
24              new LinkedBlockingDeque<Runnable>());
25
26          sce.getServletContext().setAttribute("executor", executor);
27
28          System.out.println("===== LoadSpringContext Iniciado =====");
29          LoadSpringContext.getInstance().getContext();
30      }
31
32      public void contextDestroyed(ServletContextEvent sce) {
33
34          DefaultMessageListenerContainer defaultMessageListenerContainer = (DefaultMessageListenerContainer)
35              LoadSpringContext.getInstance().getContext().getBean("jmsContainer");
36          //encerra as threads do JMS gerenciada pelo spring
37          defaultMessageListenerContainer.shutdown();
38
39          ProducerMessage producer = (ProducerMessage) LoadSpringContext.getInstance().getContext().getBean("producerMessage");
40          //fecha a conexao e a sessao com o Hornetq
41          producer.close();
42
43          ThreadPoolExecutor executor = (ThreadPoolExecutor) sce.getServletContext().getAttribute("executor");
44          //encerra o pool de threads do Executor, responsavel pelo gerenciamento assincrono das servlets
45          executor.shutdown();
46          sce.getServletContext().removeAttribute("executor");
47
48      }
49
50  }
```

Modelo de uso atual

```
13
14 public abstract class GatewayServlet extends HttpServlet {
15     private static final long serialVersionUID = -4294404416746373387L;
16
17     /** @see Processor */
18     protected abstract Processor getProcessor();
19
20     /**
21      * Utilizado caso seja necessário obter parâmetros da request, para possíveis
22      * tratamento do método getProcessor()
23      */
24     protected HttpServletRequest httpServletRequest;
25
26     public void doPost(final HttpServletRequest request, final HttpServletResponse response) {
27         request.getSession().invalidate();
28
29         final AsyncContext asyncContext = request.startAsync();
30         asyncContext.setTimeout(100 * 1000);
31         asyncContext.addListener(new CallbackAsyncListener());
32         httpServletRequest = request;
33         if(StringUtils.isBlank(request.getParameter("xml"))
34             && !request.getServletPath().contains("notificacao")) {
35             StringBuilder sb = new StringBuilder();
36             sb.append("<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>")
37             .append("<erro>")
38             .append("<codigo>412</codigo>")
39             .append("<mensagem>XML não informado</mensagem>")
40             .append("</erro>");
41             request.setAttribute("xml", sb.toString());
42             asyncContext.complete();
43             return;
44         }
45         Executor executor = (Executor) request.getServletContext().getAttribute("executor");
46         Processor processor = getProcessor();
47         processor.setSessionID(request.getSession().getId());
48         processor.setAsyncContext(asyncContext);
49         executor.execute(new RequestProcessor(processor));
50     }
51
52 }
```

Facilidade ao usar JMS

WildFly 9.0.1.Final

Messages: 0 | admin |

Configuration: Subsystems » Subsystem: Messaging » Messaging Provider: default

[Close X](#)

MESSAGING DESTINATIONS

[Queues/Topics](#) [Connection Factories](#) [Security Settings](#) [Address Settings](#) [Diverts](#)

JMS Endpoints: Provider default

Queue and Topic destinations.

[Queues](#) [Topics](#)

[Add](#) [Remove](#)

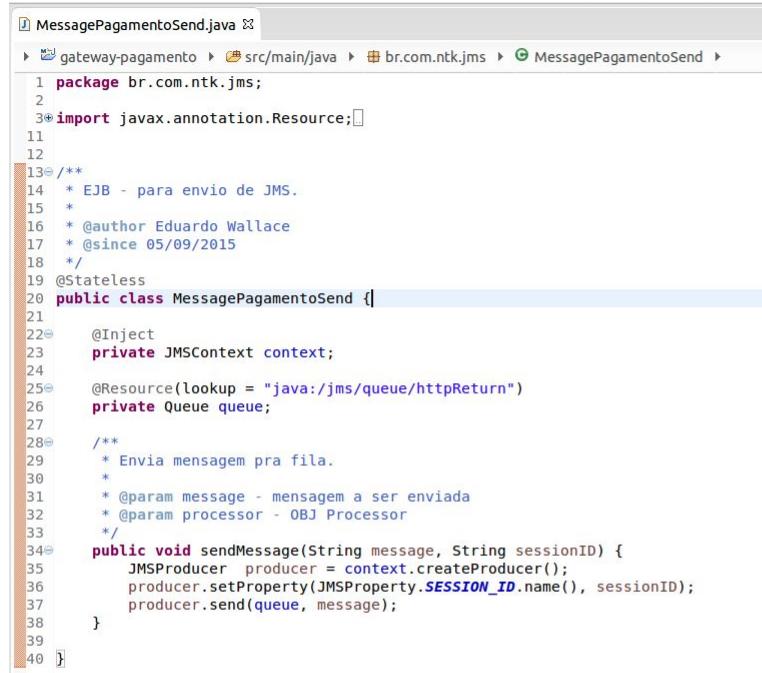
Name	JNDI
ExpiryQueue	[java:/jms/queue/ExpiryQueue]
DLQ	[java:/jms/queue/DLQ]
queueCartao	[java:/jms/queue/queueCartao]
httpReturn	[java:/jms/queue/httpReturn]
queuePagamento	[java:/jms/queue/queuePagamento]
queueBoleto	[java:/jms/queue/queueBoleto]
queueDebito	[java:/jms/queue/queueDebito]

« « 1-7 of 7 » »

[Need Help?](#)

Name:	ExpiryQueue
JNDI Names:	java:/jms/queue/ExpiryQueue
Durable?:	true
Selector:	

Modelo de uso com JEE - envio



The screenshot shows a Java code editor with the file `MessagePagamentoSend.java` open. The code is annotated with Javadoc and uses annotations like `@Stateless`, `@Inject`, and `@Resource`. The class contains a method `sendMessage` that creates a producer, sets session ID properties, and sends a message to a queue.

```
1 package br.com.ntk.jms;
2
3 import javax.annotation.Resource;
4
5
6 /**
7  * EJB - para envio de JMS.
8  *
9  * @author Eduardo Wallace
10 * @since 05/09/2015
11 */
12
13 @Stateless
14 public class MessagePagamentoSend {
15
16     @Inject
17     private JMSContext context;
18
19     @Resource(lookup = "java:/jms/queue/httpReturn")
20     private Queue queue;
21
22     /**
23      * Envia mensagem pra fila.
24      *
25      * @param message - mensagem a ser enviada
26      * @param processor - OBJ Processor
27      */
28     public void sendMessage(String message, String sessionId) {
29         JMSProducer producer = context.createProducer();
30         producer.setProperty(JMSProperty.SESSION_ID.name(), sessionId);
31         producer.send(queue, message);
32     }
33 }
34 }
```

Modelo de uso atual

```
hornetq-jms.xml
1/           <entry name="/queue/Router_1"/>
18          </queue>
19          <queue name="Pagamento_1">
20              <entry name="/queue/Pagamento_1"/>
21          </queue>
22      - - -
23
24
25          <queue name="httpReturn">
26              <entry name="/queue/httpReturn"/>
27          </queue>
28          <queue name="queueLog">
29              <entry name="/queue/queueLog"/>
30          </queue>
31          <queue name="queuePagamento">
32              <entry name="/queue/queuePagamento"/>
33          </queue>
34          <queue name="queueCartao">
35              <entry name="/queue/queueCartao"/>
36          </queue>
37          <queue name="queueBoleto">
38              <entry name="/queue/queueBoleto"/>
39          </queue>
40          <queue name="queueDebito">
41              <entry name="/queue/queueDebito"/>
42          </queue>
43          <queue name="queueSubAquirer">
44              <entry name="/queue/queueSubAquirer"/>
45          </queue>
46          <queue name="queueCheque">
47              <entry name="/queue/queueCheque"/>
48          </queue>
49          <queue name="queueAntifraude">
50              <entry name="/queue/queueAntifraude"/>
51          </queue>
```

Modelo de uso atual - envio

```
18  @Component
19  public class MessageProducerSend {
20
21      @Autowired
22      private ConnectionFactory connectionFactoryHornetQ;
23
24      private Connection connection;
25      private MessageProducer producer;
26      private Session session;
27
28      /**
29       * Setter ConnectionFactory bean TransacaoDebitoService
30       * @param connectionFactory injection container spring
31       */
32      public void setConnectionFactoryHornetQ(
33          ConnectionFactory connectionFactoryHornetQ) {
34          this.connectionFactoryHornetQ = connectionFactoryHornetQ;
35      }
36
37      private void prepareSender() throws JMSException {
38          if (connection == null) {
39              connection = connectionFactoryHornetQ.createConnection();
40          }
41          if (session == null) {
42              session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
43          }
44          if (producer == null) {
45              producer = session.createProducer(null);
46          }
47      }
48
49      private void setMessageProperties(Message message, Map<String, Object> parameters) throws JMSException {
50          //MORE CODE
51      }
52
53      public void createAndSend(String message, Map<String, Object> parameters, String queue) throws JMSException {
54          prepareSender();
55          TextMessage textMessage = session.createTextMessage();
56          setMessageProperties(textMessage, parameters);
57          textMessage.setText(message);
58          if (StringUtils.isEmpty(queue)) {
59              throw new JMSException("nao preciso informar o nome da fila");
60          }
61          producer.send(session.createQueue(queue), textMessage);
62      }
63  }
```

Configuration: Subsystems » Subsystem: EJB 3

[Close](#)

CONTAINER

BEAN POOLS

STATE MANAGEMENT

SERVICES

Bean Pools

A bean instance pool with a strict upper limit

[Add](#) [Remove](#)

Name	Pool Size
slsb-strict-max-pool	20
mdb-strict-max-pool	20

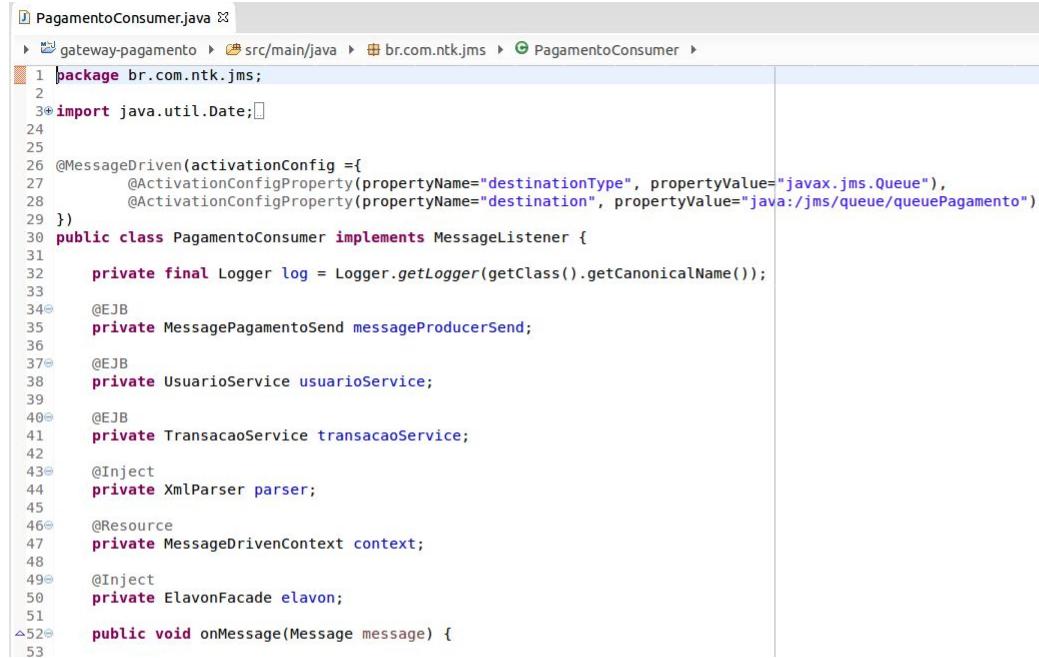
 1-2 of 2 Attributes[Need Help?](#)[Edit](#)

Max pool size: 20

Timeout: 5

Timeout unit: MINUTES

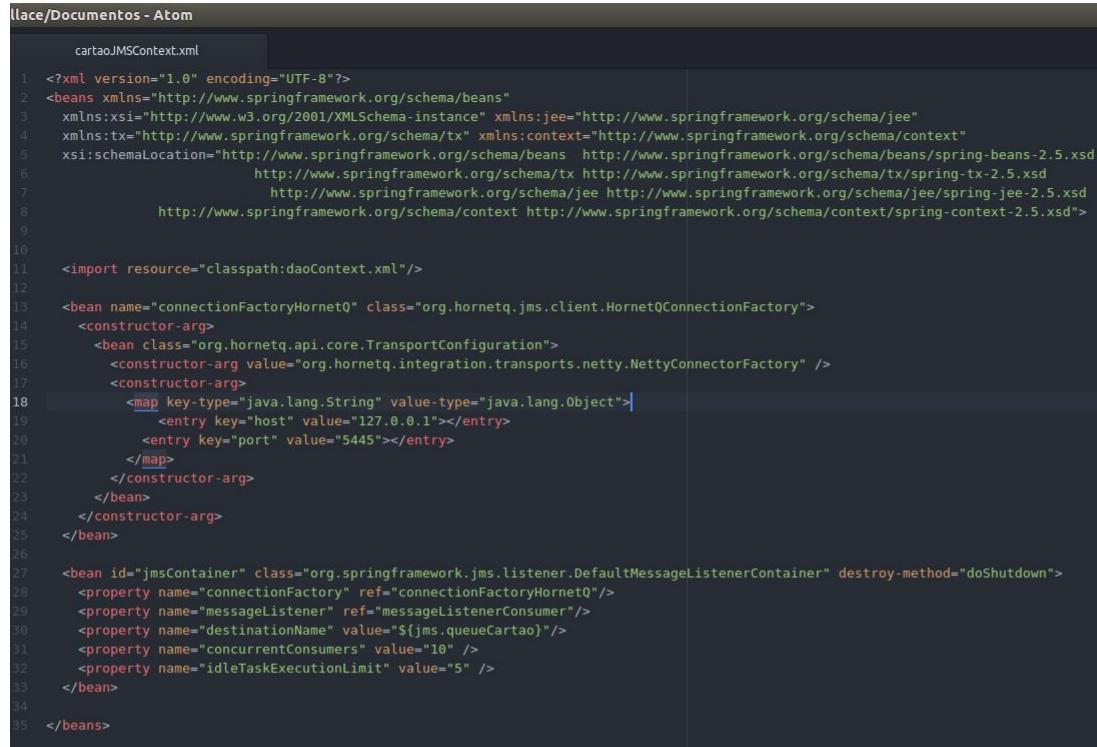
Modelo de uso com JEE - consumo



The screenshot shows a Java code editor with the file `PagamentoConsumer.java` open. The code is annotated with various JEE annotations and imports. The imports include `java.util.Date`, `@MessageDriven`, `@ActivationConfigProperty`, `@EJB`, `@Inject`, `@Resource`, and `ElavonFacade`. The class `PagamentoConsumer` implements `MessageListener` and contains a single method `onMessage`.

```
1 package br.com.ntk.jms;
2
3@import java.util.Date;
4
5
6 @MessageDriven(activationConfig ={
7     @ActivationConfigProperty(propertyName="destinationType", propertyValue="javax.jms.Queue"),
8     @ActivationConfigProperty(propertyName="destination", propertyValue="java:/jms/queue/queuePagamento")
9 })
10 public class PagamentoConsumer implements MessageListener {
11
12     private final Logger log = Logger.getLogger(getClass().getCanonicalName());
13
14     @EJB
15     private MessagePagamentoSend messageProducerSend;
16
17     @EJB
18     private UsuarioService usuarioService;
19
20     @EJB
21     private TransacaoService transacaoService;
22
23     @Inject
24     private XmlParser parser;
25
26     @Resource
27     private MessageDrivenContext context;
28
29     @Inject
30     private ElavonFacade elavon;
31
32     public void onMessage(Message message) {
33
34 }
```

Modelo de uso atual



The screenshot shows a code editor window with the title "llace/Documentos - Atom". The file being edited is "cartaoJMSContext.xml". The code is a Spring XML configuration file. It defines a connection factory for HornetQ and a JMS container.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:jee="http://www.springframework.org/schema/jee"
       xmlns:tx="http://www.springframework.org/schema/tx" xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
                           http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-2.5.xsd
                           http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-2.5.xsd
                           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-2.5.xsd">

    <import resource="classpath:daoContext.xml"/>

    <bean name="connectionFactoryHornetQ" class="org.hornetq.jms.client.HornetQConnectionFactory">
        <constructor-arg>
            <bean class="org.hornetq.api.core.TransportConfiguration">
                <constructor-arg value="org.hornetq.integration.transports.netty.NettyConnectorFactory" />
                <constructor-arg>
                    <map key-type="java.lang.String" value-type="java.lang.Object">
                        <entry key="host" value="127.0.0.1"/>
                        <entry key="port" value="5445"/>
                    </map>
                </constructor-arg>
            </bean>
        </constructor-arg>
    </bean>

    <bean id="jmsContainer" class="org.springframework.jms.listener.DefaultMessageListenerContainer" destroy-method="doShutdown">
        <property name="connectionFactory" ref="connectionFactoryHornetQ"/>
        <property name="messageListener" ref="messageListenerConsumer"/>
        <property name="destinationName" value="${jms.queueCartao}"/>
        <property name="concurrentConsumers" value="10" />
        <property name="idleTaskExecutionLimit" value="5" />
    </bean>
</beans>
```

Modelo de uso atual - consumo

```
50  @Component
51  public class MessageListenerConsumer implements MessageListener {
52
53      private static Logger LOGGER = LogManager.getLogger(MessageListenerConsumer.class);
54
55      @Autowired
56      TransacaoCartaoService transacaoCartaoService;
57
58      @Autowired
59      MessageProducerSend producer;
60
61      public void onMessage(Message message) {
62
```

Facilidade em usar Factory

WildFly 9.0.1.Final | Messages: 0 | admin | 🔍

Configuration: Subsystems » Subsystem: EJB 3 [Close](#)

CONTAINER BEAN POOLS STATE MANAGEMENT SERVICES

Container Thread Pools Remoting Profiles

Thread Pools

A thread pool executor with an unbounded queue. Such a thread pool has a core size and a queue with no upper bound. When a task is submitted, if the number of running threads is less than the core size, a new thread is created. Otherwise, the task is placed in queue. If too many tasks are allowed to be submitted to this type of executor, an out of memory condition may occur.

Add Remove

Name	Max Threads
default	10

« < 1-1 of 1 > »

Attributes

Edit Need Help?

Max threads: 10

Name: default

Thread factory:

Modelo de uso com JEE



The screenshot shows a Java code editor with the file `CallbackContext.java` open. The code is annotated with comments explaining its purpose and usage.

```
15 *  
16 * EJB Singleton, utilizado para controlar o armazenamento e remocao dos Callbacks.  
17 *  
18 * @author Eduardo Wallace  
19 * @since 05/09/2015  
20 */  
21 @Startup  
22 @Singleton  
23 @ConcurrencyManagement(ConcurrencyManagementType.CONTAINER)  
24 public class CallbackContext {  
25  
26     private final Logger log = Logger.getLogger(getClass().getCanonicalName());  
27  
28     private ConcurrentHashMap<String, AsyncContext> mapRequests;  
29  
30     @PostConstruct  
31     private void init() {  
32         log.info("Singleton EJB CallbackContext - Iniciado!");  
33         mapRequests = new ConcurrentHashMap<String, AsyncContext>();  
34     }  
35  
36     /** Armazena o AsyncContext **/  
37     public void put(String sessionID, AsyncContext asyncContext) {  
38         mapRequests.put(sessionID, asyncContext);  
39     }  
40  
41     /** Remove o AsyncContext **/  
42     public void remove(String sessionID) {  
43         mapRequests.remove(sessionID);  
44     }  
45  
46     /**  
47     * Conclui o processamento da transacao do gateway chamando o Callback  
48     *  
49     * @param xml - contem os dados da transacao  
50     * @param sessionID - id da sessao  
51     */  
52     @Lock(LockType.WRITE)  
53     public void complete(String xml, String sessionID) {  
54         if (mapRequests.containsKey(sessionID)) {  
55             AsyncContext asyncContext = (AsyncContext) mapRequests.get(sessionID);  
56             mapRequests.remove(sessionID);  
57             asyncContext.getRequest().setAttribute("xml", xml);  
58             asyncContext.complete();  
59         } else {  
60             log.warning("Timeout " + xml);  
61         }  
62     }  
63 }  
64 //
```

Modelo de uso atual

```
StartCartao.java
1 package br.com.gateway.cartao.util;
2
3
4
5 /**
6 *
7 * @author Luis Requejo
8 * @author NTK Solutions
9 *
10 */
11 public class StartCartao {
12
13     public static void main(String[] args) {
14         LoadSpringContext.getInstance().getContext();
15         try {
16             XMLParse.getInstance();
17         } catch (Exception e) {
18             System.out.println("Erro na criacao dos contextos JAXB");
19             System.exit(0);
20         }
21         System.out.println("Modulo Cartao Iniciado");
22     }
23 }
24
```

Modelo de uso atual

```
LoadSpringContext.java
1 import org.springframework.context.ApplicationContext;
2 import org.springframework.context.support.ClassPathXmlApplicationContext;
3
4 public class LoadSpringContext {
5
6     private static LoadSpringContext springContext = null;
7     private static final ConcurrentHashMap<String, AsyncContext> mapRequests;
8
9     private final ApplicationContext context;
10
11    private LoadSpringContext() {
12        context = new ClassPathXmlApplicationContext("jmsContext.xml");
13    }
14
15    public static LoadSpringContext getInstance(){
16        if (springContext == null) {
17            springContext = new LoadSpringContext();
18        }
19        return springContext;
20    }
21
22    public ApplicationContext getContext() {
23        return context;
24    }
25
26    static{
27        mapRequests = new ConcurrentHashMap<String, AsyncContext>();
28    }
29
30    public ConcurrentHashMap<String, AsyncContext> getMapRequests() {
31        return mapRequests;
32    }
33
34    /**
35     * Conclui o processamento da transacao do gateway
36     *
37     * @param xml - contem os dados da transacao
38     * @param sessionID - id da sessao
39     */
40    public void complete(String xml, String sessionID) {
41        if(mapRequests.containsKey(sessionID)){
42            AsyncContext asyncContext = (AsyncContext) mapRequests.get(sessionID);
43            synchronized (asyncContext) {
44                mapRequests.remove(sessionID);
45                asyncContext.getRequest().setAttribute("xml", xml);
46                asyncContext.complete();
47            }
48        }else{
49            System.out.println("time out "+xml);
50        }
51    }
52}
```

WildFly 9.0.1.Final | Messages: 0 | admin |

Server: Standalone Server » Monitor: Log Files Close

LOG FILES

Log Viewer
Log files of selected server

Filter: [Download](#) [View](#)

▲ Log File Name	Date - Time (UTC)	Size (MB)
server.log	2015-09-18T13:41:53.000-0300	0.05
server.log.2015-09-04	2015-09-04T19:13:05.000-0300	1.67
server.log.2015-09-05	2015-09-05T13:56:21.000-0300	0.04
server.log.2015-09-06	2015-09-06T17:19:36.000-0300	0.11
server.log.2015-09-07	2015-09-07T22:20:37.000-0300	0.72
server.log.2015-09-08	2015-09-08T23:45:58.000-0300	1.61
server.log.2015-09-09	2015-09-09T16:02:15.000-0300	0.04
server.log.2015-09-11	2015-09-11T18:41:03.000-0300	0.20
server.log.2015-09-14	2015-09-14T19:06:13.000-0300	1.08
server.log.2015-09-17	2015-09-17T18:27:10.000-0300	28.19

« < 1-10 of 10 > »

LOG FILES

SERVER...-09-05

server.log.2015-09-05

Find



```
387 2015-09-05 13:46:00,355 INFO [org.jboss.weld.deployer] (MSC service thread 1-2) WFLYWELD0009: Starting weld service for deployment gateway-1
388 2015-09-05 13:46:00,421 INFO [org.jboss.as.ejb3] (MSC service thread 1-8) WFLYEJB0042: Started message driven bean 'MessageListenerConsumer'
389 2015-09-05 13:46:05,182 INFO [br.com.ntk.listener.CallbackContext] (ServerService Thread Pool -- 64) Singleton EJB CallbackContext - Iniciac
390 2015-09-05 13:46:10,737 INFO [stdout] (ServerService Thread Pool -- 64) ===== LoadContext Iniciado =====
391 2015-09-05 13:46:10,856 INFO [javax.enterprise.resource.webcontainer.jsf.config] (ServerService Thread Pool -- 64) Inicializando Mojarra 2.2
392 2015-09-05 13:46:11,476 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 64) WFLYUT0021: Registered web context: /gateway
393 2015-09-05 13:46:11,545 INFO [org.jboss.as.server] (ServerService Thread Pool -- 37) WFLYSRV010: Deployed "gateway-httpRoteador.war" (runti
394 2015-09-05 13:46:11,696 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://127.0.0.1:95
395 2015-09-05 13:46:11,697 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
396 2015-09-05 13:46:11,697 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: WildFly Full 9.0.1.Final (WildFly Core 1.0.1.Final) starte
397 2015-09-05 13:50:23,298 INFO [br.com.ntk.servlet.RequestProcessor] (EE-ManagedExecutorService-default-Thread-1) <transacao>
398    <usuario>ntk</usuario>
399    <token>1234</token>
400    <id_transacao>932f7234-9352-4e84-9799-d407d35b6949</id_transacao>
401  </transacao>
402 2015-09-05 13:50:23,518 INFO [stdout] (default task-3) 3SwdVBQGZVjzDMLiI3yGhzB3NjbCVlwap8YaKhw
403 2015-09-05 13:50:24,551 INFO [br.com.ntk.servlet.RequestProcessor] (EE-ManagedExecutorService-default-Thread-2) <transacao>
404    <usuario>ntk</usuario>
405    <token>1234</token>
406    <id_transacao>932f7234-9352-4e84-9799-d407d35b6949</id_transacao>
407  </transacao>
408 2015-09-05 13:50:24,580 INFO [stdout] (default task-5) BfRm1T6Gi0xKw1_4-mYRMQjTp7C3002dtXdLq-
409 2015-09-05 13:50:25,231 INFO [br.com.ntk.servlet.RequestProcessor] (EE-ManagedExecutorService-default-Thread-3) <transacao>
410    <usuario>ntk</usuario>
411    <token>1234</token>
412    <id_transacao>932f7234-9352-4e84-9799-d407d35b6949</id_transacao>
413  </transacao>
414 2015-09-05 13:50:25,263 INFO [stdout] (default task-7) lhyAfgKU4L12qdDSeqFZ4MIuSSbgY3gZF2oAcK7n
415
```

Modelo atual dos Logs

```
E-commerce:/usr/src/java/cartao/log/2015-09
[root@E-commerce log]# ls
2015-01  2015-02  2015-03  2015-04  2015-05  2015-06  2015-07  2015-09 _application.log  application.log  bkp  error.log
[root@E-commerce log]# cd 2015-09
[root@E-commerce 2015-09]# ls
2015-09-03-1.log  2015-09-08-1.log  2015-09-10-1.log  2015-09-11-1.log  2015-09-14-1.log  2015-09-15-1.log
[root@E-commerce 2015-09]#
```

Modelo atual dos Logs

```
18/09/2015 13:46:00.066 | DEBUG | b.c.n.u.EmailUtil:50 | Destinatarios: [eduardo.silva@ntk-consult.com.br]
com.amazonaws.AmazonServiceException: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: c0ee89c4-5e24-11e5-be3e-8b07a1f570e6)
    at com.amazonaws.http.AmazonHttpClient.handleErrorResponse(AmazonHttpClient.java:1160)
    at com.amazonaws.http.AmazonHttpClient.executeOneRequest(AmazonHttpClient.java:748)
    at com.amazonaws.http.AmazonHttpClient.executeHelper(AmazonHttpClient.java:467)
    at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:302)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.invoke(AmazonSimpleEmailServiceClient.java:1731)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.sendEmail(AmazonSimpleEmailServiceClient.java:1461)
    at br.com.ntk.util.EmailUtil.sendAnexo(EmailUtil.java:78)
    at br.com.ntk.schedule.ReenvioQuartzJob.execute(ReenvioQuartzJob.java:36)
    at org.quartz.core.JobRunShell.run(JobRunShell.java:202)
    at org.quartz.simpl.SimpleThreadPool$WorkerThread.run(SimpleThreadPool.java:573)
18/09/2015 13:46:01.429 | ERROR | b.c.n.u.EmailUtil:90 | ERRO AO ENVIAR O EMAIL: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: 08430920-5e25-11e5-b998-fd4074f4c6a0)
18/09/2015 13:46:01.431 | DEBUG | b.c.n.d.EmailDAO:48 | ID_Mensagem: null atualizado para email 1068 com sucesso!
18/09/2015 13:46:01.433 | DEBUG | b.c.n.d.EmailDAO:60 | Destinatario: eduardo.silva@ntk-consult.com.br Status: 1 atualizado!
18/09/2015 13:48:00.003 | DEBUG | b.c.n.s.ReenvioQuartzJob:24 | JobExecution
18/09/2015 13:48:00.004 | DEBUG | b.c.n.s.ReenvioQuartzJob:31 | Email para retentativa encontrado: br.com.ntk.model.Email@bfa683
18/09/2015 13:48:00.004 | DEBUG | b.c.n.u.EmailUtil:49 | Remetente: null
18/09/2015 13:48:00.004 | DEBUG | b.c.n.u.EmailUtil:50 | Destinatarios: [eduardo.silva@ntk-consult.com.br]
com.amazonaws.AmazonServiceException: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: 08430920-5e25-11e5-b998-fd4074f4c6a0)
    at com.amazonaws.http.AmazonHttpClient.handleErrorResponse(AmazonHttpClient.java:1160)
    at com.amazonaws.http.AmazonHttpClient.executeOneRequest(AmazonHttpClient.java:748)
    at com.amazonaws.http.AmazonHttpClient.executeHelper(AmazonHttpClient.java:467)
    at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:302)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.invoke(AmazonSimpleEmailServiceClient.java:1731)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.sendEmail(AmazonSimpleEmailServiceClient.java:1461)
    at br.com.ntk.util.EmailUtil.sendAnexo(EmailUtil.java:78)
    at br.com.ntk.schedule.ReenvioQuartzJob.execute(ReenvioQuartzJob.java:36)
    at org.quartz.core.JobRunShell.run(JobRunShell.java:202)
    at org.quartz.simpl.SimpleThreadPool$WorkerThread.run(SimpleThreadPool.java:573)
18/09/2015 13:48:01.097 | ERROR | b.c.n.u.EmailUtil:90 | ERRO AO ENVIAR O EMAIL: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: 08430920-5e25-11e5-b998-fd4074f4c6a0)
18/09/2015 13:48:01.099 | DEBUG | b.c.n.d.EmailDAO:48 | ID_Mensagem: null atualizado para email 1068 com sucesso!
18/09/2015 13:48:01.101 | DEBUG | b.c.n.d.EmailDAO:60 | Destinatario: eduardo.silva@ntk-consult.com.br Status: 1 atualizado!
18/09/2015 13:50:00.005 | DEBUG | b.c.n.s.ReenvioQuartzJob:24 | JobExecution
18/09/2015 13:50:00.005 | DEBUG | b.c.n.s.ReenvioQuartzJob:31 | Email para retentativa encontrado: br.com.ntk.model.Email@bfa683
18/09/2015 13:50:00.005 | DEBUG | b.c.n.u.EmailUtil:49 | Remetente: null
18/09/2015 13:50:00.006 | DEBUG | b.c.n.u.EmailUtil:50 | Destinatarios: [eduardo.silva@ntk-consult.com.br]
com.amazonaws.AmazonServiceException: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: 5001e3b6-5e25-11e5-bcec-1ba2018ca6d8)
    at com.amazonaws.http.AmazonHttpClient.handleErrorResponse(AmazonHttpClient.java:1160)
    at com.amazonaws.http.AmazonHttpClient.executeOneRequest(AmazonHttpClient.java:748)
    at com.amazonaws.http.AmazonHttpClient.executeHelper(AmazonHttpClient.java:467)
    at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:302)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.invoke(AmazonSimpleEmailServiceClient.java:1731)
    at com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClient.sendEmail(AmazonSimpleEmailServiceClient.java:1461)
    at br.com.ntk.util.EmailUtil.sendAnexo(EmailUtil.java:78)
    at br.com.ntk.schedule.ReenvioQuartzJob.execute(ReenvioQuartzJob.java:36)
    at org.quartz.core.JobRunShell.run(JobRunShell.java:202)
    at org.quartz.simpl.SimpleThreadPool$WorkerThread.run(SimpleThreadPool.java:573)
18/09/2015 13:50:01.485 | ERROR | b.c.n.u.EmailUtil:90 | ERRO AO ENVIAR O EMAIL: 1 validation error detected: Value null at 'source' failed to satisfy constraint: Member must not be null (Service: AmazonSimpleEmailService; Status Code: 400; Error Code: ValidationException; Request ID: 5001e3b6-5e25-11e5-bcec-1ba2018ca6d8)
18/09/2015 13:50:01.487 | DEBUG | b.c.n.d.EmailDAO:48 | ID_Mensagem: null atualizado para email 1068 com sucesso!
18/09/2015 13:50:01.489 | DEBUG | b.c.n.d.EmailDAO:60 | Destinatario: eduardo.silva@ntk-consult.com.br Status: 1 atualizado!
```

WildFly 9.0.1.Final

Messages: 0 | admin | [Logout](#)

Configuration: Subsystems » Subsystem: Datasources [Close X](#)

DATASOURCES XA DATA SOURCES

JDBC Datasources

JDBC datasource configurations.

Add Remove Disable

Name	JNDI	Enabled?
ExampleDS	java:jboss/datasources/ExampleDS	✓
gatewayDS	java:jboss/datasources/gatewayDS	✓

« « 1-2 of 2 » »

Attributes Connection Pool Security Properties Validation Timeouts Statements

Flush Gracefully Need Help?

[Edit](#)

Min Pool Size: 10

Initial Pool Size:

Max Pool Size: 30

Prefill: true

Flush Strategy:

Strict Minimum: false

Use Fast Fail: false

Gerenciamento Datasource

WildFly 9.0.1.Final

Server: Standalone Server » Monitor: Subsystems » Subsystem: Datasources

Close

DATA SOURCES XA DATA SOURCES

Test Connection Flush Gracefully ▾

Name	JNDI	Enabled?	Statistics Enabled?
ExampleDS	java:jboss/datasources/ExampleDS	true	false
gatewayDS	java:jboss/datasources/gatewayDS	true	false

« « 1-2 of 2 » »

Refresh Results

Connection Pool

Available Connections 0
Active 0
Max Used 0

Active total number

Max Used total number

Prepared Statement Cache

Access Count 0
Hit Count 0
Miss Count 0

Hit Count total number

Miss Count total number

```
persistencia.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2<persistence version="2.1"
3   xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
5<persistence-unit name="gatewayPU" transaction-type="JTA">
6
7   <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>
8   <jta-data-source>java:jboss/datasources/gatewayDS</jta-data-source>
9
10  <class>br.com.gateway.entity.Usuario</class>
11  <class>br.com.gateway.entity.Configuracao</class>
12  <class>br.com.ntk.model.Transacao</class>
13
14<properties>
15   <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL5InnoDBDialect" />
16   <property name="hibernate.hbm2ddl.auto" value="update" /> <!-- create-drop -->
17   <property name="hibernate.connection.charSet" value="UTF-8" />
18   <property name="hibernate.show_sql" value="true" />
19   <property name="hibernate.format_sql" value="true" />
20 </properties>
21
22 </persistence-unit>
23 </persistence>
```

Modelo atual Datasource

```
daoContext.xml

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:jee="http://www.springframework.org/schema/jee"
4   xmlns:tx="http://www.springframework.org/schema/tx" xmlns:context="http://www.springframework.org/schema/context"
5   xmlns:util="http://www.springframework.org/schema/util"
6   xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
7           http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-2.5.xsd
8           http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-2.5.xsd
9           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-2.5.xsd
10          http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-3.0.xsd">
11
12 <util:properties id="applicationProps" location="file:${propConfig}" />
13 <context:property-placeholder properties-ref="applicationProps" />
14
15     <!-- Gerenciador de Transações -->
16     <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager" >
17         <property name="dataSource" ref="dataSource"/>
18     </bean>
19
20     <!-- Habilita o comportamento transacional onde houver a anotação @Transacional -->
21     <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
22         <property name="driverClassName" value="${jdbc.driverClassName}"/>
23         <property name="url" value="${jdbc.url}"/>
24         <property name="username" value="${jdbc.username}"/>
25         <property name="password" value="${jdbc.password}"/>
26         <property name="validationQuery" value="SELECT 1"/>
27         <property name="initialSize" value="5" />
28         <property name="maxActive" value="10" />
29     </bean>
30
31     <tx:annotation-driven transaction-manager="transactionManager" />
32
33     <!-- Dar Suporte à Injeção de Dependência do Bean via Annotations -->
34     <context:annotation-config />
35
36     <context:component-scan base-package="br.com.gateway" annotation-config="true" />
37
38     <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
39         <property name="dataSource" ref="dataSource" />
40     </bean>
41
42 </beans>
```

WildFly 9.0.1.Final

Messages: 0 | admin | 🔍

Home Deployments Configuration Runtime Access Control Patching

Deployment	Add	Nested Deployment
gatewayEAR.ear	▶	gatewayPagamento... gateway-pagamento... gateway-dao.jar

Deployment

The deployment gatewayEAR.ear is enabled and has the status OK.

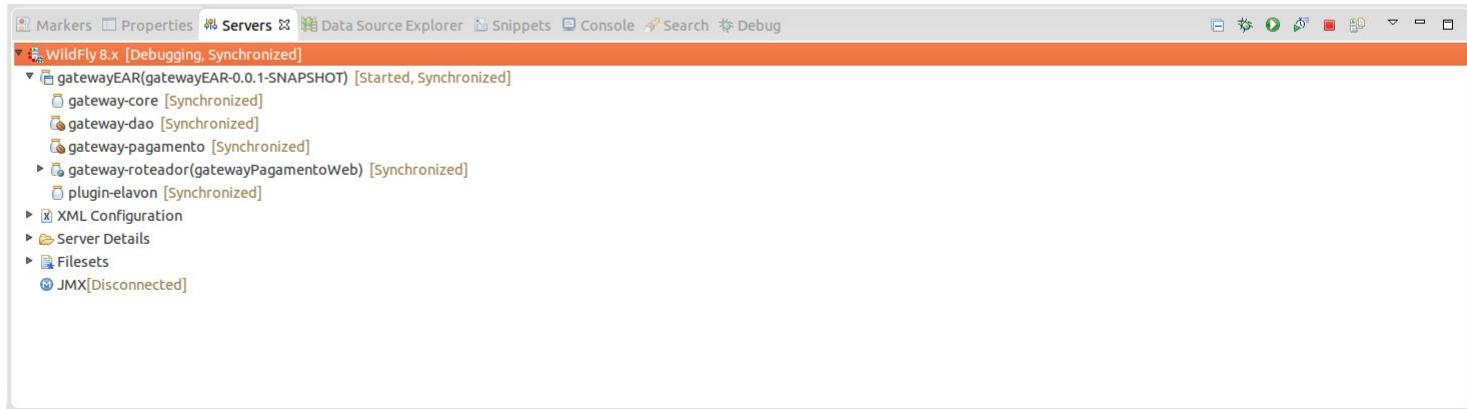
Details

- Last enabled at 2015-09-18 13:40:07,979 BRT
- The deployment was never disabled
- Runtime name: gatewayEAR.ear

Nested Deployments

The deployment contains 3 nested deployments

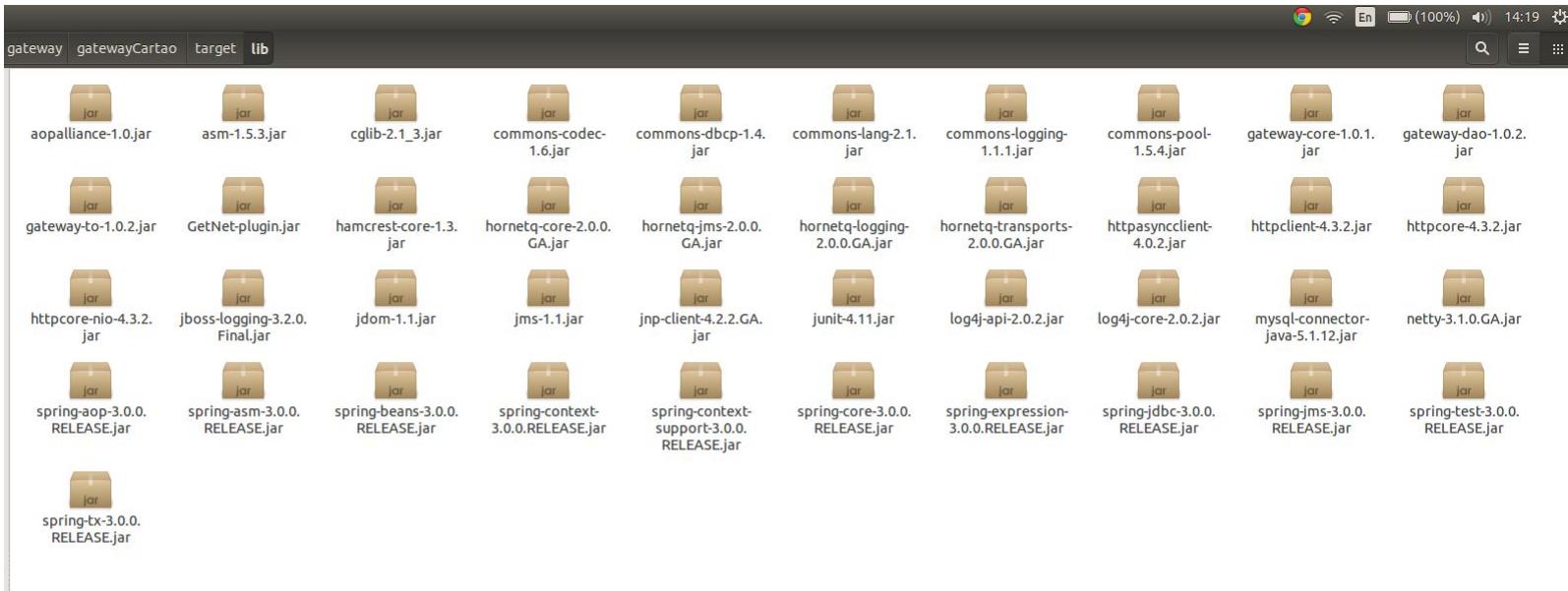
Facilidade no Deploy



Modelo atual para cada modulo - Deploy

```
root@E-commerce:/usr/src/java/cartao          x  wallace@wallace:~  
[root@E-commerce cartao]# ls  
gateway-cartao-2.1.0.jar  getnet  init.sh  lib  log  
[root@E-commerce cartao]# cat init.sh  
#!/bin/sh  
  
pid=`ps -ef | grep gatewayCartao | grep -v grep| awk -F' ' '{print $2}'`;  
  
if test "$pid" = "" ; then  
/usr/src/java/jdk1.7.0/bin/java -Dlog4j.path=/usr/src/java/cartao/log/ -Dlog4j.configurationFile=/usr/src/java/conf/log4j2.xml -DpropConfig=/usr/src/java/conf/conf.properties -jar /usr/src/java/cartao/gatewa  
y-cartao-*.jar &  
fi  
[root@E-commerce cartao]#
```

Modelo atual para cada modulo - Deploy



Modelo atual para cada modulo - Deploy

```
E-commerce:/usr/src/java/apache-tomcat-7.0.20/webapps/httpRoteador/WEB-INF/lib
[root@E-commerce webapps]# ls
EmailService EmailService.war gateway gateway.war gatewaywsdl gatewaywsdl.war httpRoteador httpRoteador.war
[root@E-commerce webapps]# cd httpRoteador
[root@E-commerce httpRoteador]# ls
check.jsp index.jsp META-INF WEB-INF
[root@E-commerce httpRoteador]# cd WEB-INF/lib/
[root@E-commerce lib]# ls
aopalliance-1.0.jar    hornetq-core-2.0.0.GA.jar      httpclient-4.4.jar        jnp-client-4.2.2.GA.jar   netty-3.1.0.GA.jar      spring-context-support-3.0.0.RELEASE.jar
commons-codec-1.9.jar   hornetq-jms-2.0.0.GA.jar     httpcore-4.4.jar       json-20090211.jar    spring-aop-3.0.0.RELEASE.jar  spring-core-3.0.0.RELEASE.jar
commons-lang-2.1.jar    hornetq-logging-2.0.0.GA.jar   httpcore-nio-4.3.2.jar log4j-api-2.0.2.jar   spring-asn-3.0.0.RELEASE.jar  spring-expression-3.0.0.RELEASE.jar
commons-logging-1.1.1.jar hornetq-transports-2.0.0.GA.jar jboss-logging-3.2.0.Final.jar log4j-core-2.0.2.jar  spring-beans-3.0.0.RELEASE.jar  spring-jms-3.0.0.RELEASE.jar
gateway-common-1.0.2.jar httpasyncclient-4.0.2.jar     jms-1.1.jar          log4j-web-2.0.2.jar   spring-context-3.0.0.RELEASE.jar  spring-tx-3.0.0.RELEASE.jar
[root@E-commerce lib]#
```

VIRTUAL MACHINE STATUS

Java HotSpot(TM) 64-Bit Server VM

[Refresh Results](#)

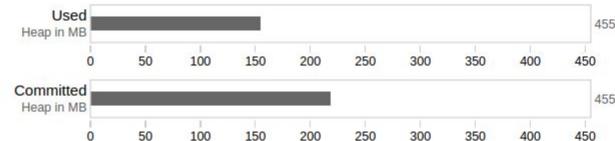
Operating System: Linux 3.16.0-49-generic

Processors: 4

JVM Uptime: 48 min, 51 s

Heap Usage

Max	455
Used	155
Committed	218



Thread Usage

Live	101
Daemon	20



JPA

[Back](#)[Basic Metrics](#)

Persistence Unit Metrics: gatewayEAR.ear/gateway-dao.jar#gatewayPU

[Refresh Results](#)

Runtime Information about Hibernate use in the deployment.

Sessions

Sessions opened 0

Closed 0

Closed

total number



Transactions

Completed 0

Successful 0

Successful

total number



Query Execution

Execution Count 0

Max Time 0

Execution Count

total number



Max Time

total number



WildFly 9.0.1.Final | Messages: 5 | admin |

Server: Standalone Server » Monitor: Subsystems

Messaging Statistics

Successfully refreshed metrics for queue httpReturn × Close

JMS Queue Metrics: Provider 'default'

Metrics for JMS queues.

Flush

Back Queues Topics

Name	JNDI
ExpiryQueue	[java:/jms/queue/ExpiryQueue]
DLQ	[java:/jms/queue/DLQ]
queueCartao	[java:/jms/queue/queueCartao]
httpReturn	[java:/jms/queue/httpReturn]
queuePagamento	[java:/jms/queue/queuePagamento]

« < 1-5 of 7 > »

Queue Metrics

Consumer Count 15 Refresh Results

Message Count 0

Messages Added 0

Scheduled Count 0