ActiveMQ 操作笔记

1. **准备工作**
   1. **本地搭建**

**安装略。**

1. **Spring中的配置**

**2.1 maven配置**

<!--activeMQ-->  
<dependency>  
 <groupId>org.apache.activemq</groupId>  
 <artifactId>activemq-all</artifactId>  
</dependency>  
<dependency>  
 <groupId>org.apache.activemq</groupId>  
 <artifactId>activemq-pool</artifactId>  
</dependency>

**2.2 spring中的配置**

生产者：



消费者：

****

1. **生产者与消费者**

**3.1 生产者**

定义抽象类JmsBaseProducerImpl 其中有抽象方法getDestination（）

实现的方法 boolean send(final String message) throws JMSException



@Resource  
private JmsTemplate jmsTemplate;  
  
public abstract Queue getDestination();  
  
@Override  
public boolean send(final String message) throws JMSException {  
 boolean result = true;  
 if (null == getDestination()) {  
 *LOG*.error("JmsBaseProducerImpl's destination is null !");  
 result = false;  
 } else {  
 try {  
 jmsTemplate.send(getDestination(), new MessageCreator() {  
 public Message createMessage(Session session)  
 throws JMSException {  
 return session.createTextMessage(message);  
 }  
 });  
 } catch (Exception e) {  
 result = false;  
 *LOG*.error("JmsBaseProducerImpl Exception:{}", e);  
 throw new JMSException(e.getMessage());  
 }  
 }  
 return result;  
}

设置指定队列的操作类 ActionProducer 继承 JmsBaseProducerImpl

*/\*\*  
 \* 指定xml中配置的队列bean  
 \*/*@Resource(name = "actionQueue")  
private Queue destination;  
  
@Override  
public Queue getDestination() {  
 return destination;  
}



具体操作时调用send方法

@Resource  
private ActionProducer actionProducer;  
  
@RequestMapping(value = "/insertDocs")  
@ResponseBody  
public ResponseVo insertDocs() {  
 ResponseVo responseVo = new ResponseVo();  
 try {  
 actionProducer.send(createTmpDoc());  
 }catch(Exception e){  
 *LOG*.error("保存数据失败:{}", e);  
 }  
  
 return responseVo;  
}



**3.2 消费者**

需要继承DefaultMessageListenerContainer类

****

**3.3 监听器**

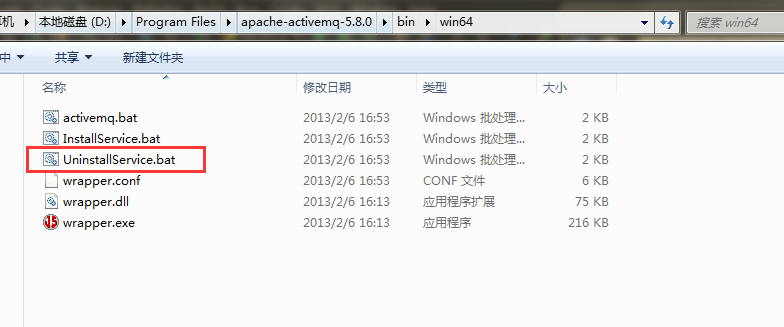
需要实现MessageListener接口

****

1. **常见问题**

**4.1 无法删除已存在的数据**

系统启动时默认开始了mq的服务，如果要删除数据 需要停止其服务



**4.2 中文乱码**

消息体为中文时，可能解析成ascii码

原因是mq的web.xml同样需要配置filter

http://blog.csdn.net/daolingren/article/details/51773137