Ejercicio 1

Creamos la clase *MsgConexion* que será la clase que usaremos para encapsular la problemática relativa a la factoría, conexión y sesión. A continuación declaramos las variables necesarias e importamos los paquetes javax.jms,* y javax.naming.*.

```
import javax.jms.*;
import javax.naming.*;

public class MsgConexion {

   public String nombreCola = "miCola";                  // Nombre externo de la cola
   public Context contexto = null;                       // Contexto JNDI
   public QueueConnectionFactory factoria = null;        // Factoria de conexiones
   public QueueConnection conexionCola = null;           // conexion
   public QueueSession sesionCola = null;                // sesion
   public Queue cola = null;                             // cola de mensajes
```

Posteriormente, crearemos el método inicializaCola que elevará la excepción JMSException y que retornara un booleano (indicando si acaba bien o mal). Si se produce un error la variable contexto quedara a null.

```
public MsgConexion() {

   }

   public boolean inicializaCola() {
      try {

         if (contexto == null) {
            // Aun no se ha realizado la inicializacion. Una vez realizada
            // no tiene sentido volver a realizarla.
            contexto = new InitialContext();          // Obtiene contexto JNDI
            // Obtiene factoria de conexion a colas (ha debido ser creada externamente)
            factoria = (QueueConnectionFactory) contexto.lookup("QueueConnectionFactory");
            // Obtiene la cola (ha debido ser creada externamente)
            cola = (Queue) contexto.lookup(nombreCola);
            // Ahora crea la conexion a la cola
            conexionCola = factoria.createQueueConnection();
            // Crea la sesion
            sesionCola = conexionCola.createQueueSession(false, Session.AUTO_ACKNOWLEDGE);
            conexionCola.start();                     // Hay que activar la conexion para empezar.
         }
      } catch (NamingException | JMSException ex) {
         contexto = null;
         return false;
      }
      return true;
```

```
    }

    public void cerrarConexion() {
        try {
            conexionCola.stop();
            conexionCola.close();
        } catch (JMSException ex) {
            System.out.println("Excepcion");
        }
    }
```

Con el método *lookup* localizamos la factoría de conexiones de colas. Una vez obtenida la factoría usamos el método *crateQueueConection* para crear la conexión y desde ella, la sesión con el método *createQueueSesion*. Con el método *cerrarConexion* las conexiones serán detenidos con el método *stop* y cerradas al concluir la aplicación con el método *close*.

Crearemos la clase Productor que contara con el método *main* para poder ser ejecutado. El emisor se crea con *createSender* y el mensaje se crea con el método *createMesage*.

```
public class Productor {

    public static void main(String[] args) throws JMSException {
        int i;                          // Para enviar varios mensajes
        MsgConexion mc;                 // Conexion propia a cola
        QueueSender emisor;             // Emisor
        TextMessage m;                  // Mensaje a enviar
        boolean ok;                     // Para comprobar retorno de metodo
        mc = new MsgConexion();         // Crea objeto MsgConexion
        ok = mc.inicializaCola();       // Prepara los parametros
        if (ok) {
                                        // Prepara emisor
            emisor = mc.sesionCola.createSender(mc.cola);
                                        // Prepara mensaje
            m = mc.sesionCola.createTextMessage(); // Crea mensaje
                                        // Hace varios envios
            for (i = 0; i < 5; i++) {
                m.setText("Hola Mundo" + i);    // Contenido mensaje
                emisor.send(m);         // ENVIO MENSAJE
            }
        }
    }
}
```

Análogamente se crea la clase *Consumidor*. La primera parte es idéntica a la del productor. Después se crea el receptor con el método *createReceiver*. A parar de ahí puede recibir mensajes mediante el método *receive* del receptor, indicándole como argumento la espera máxima en milisegundos.

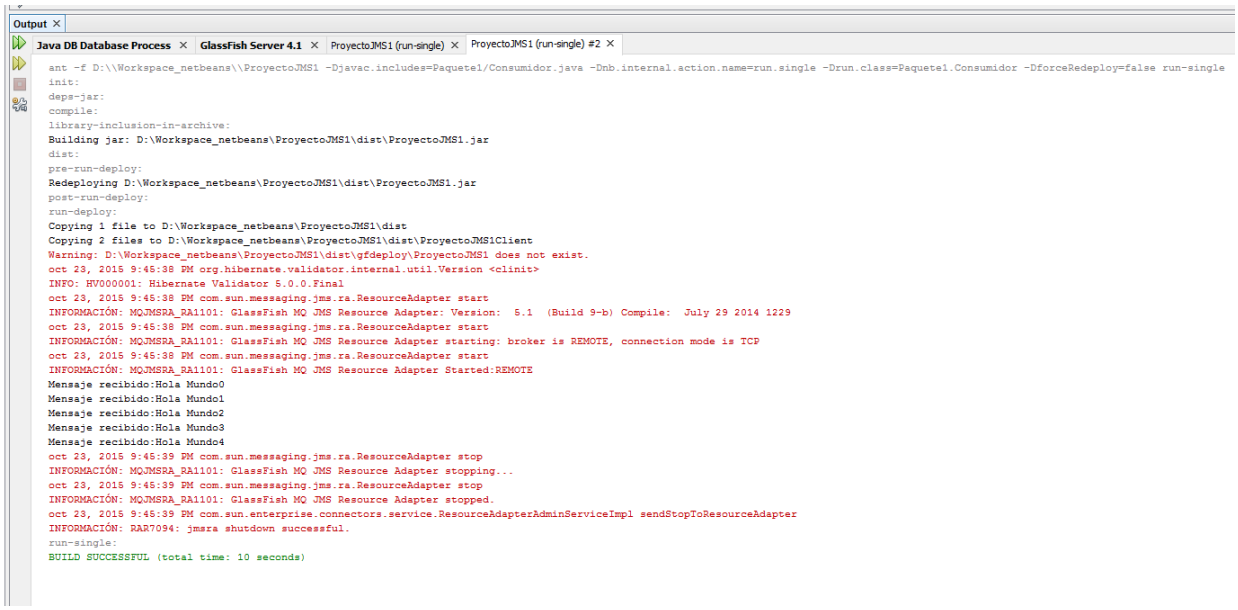
```
public class Consumidor {

   public static void main(String[] args) {
      int i;                                    // Para recibir varios mensajes
      MsgConexion mc;                           // Conexion propia a cola
      QueueReceiver receptor;                   // Receptor
      TextMessage m;                            // Mensaje recibido
      boolean ok;                               // Comprobacion retorno
      mc = new MsgConexion();                   // Crea su objeto MsgConexion
       ok = mc.inicializaCola();                // Inicializa parametros

      try {
         if (ok) {
            // Prepara receptor sobre cola
            receptor = mc.sesionCola.createReceiver(mc.cola);
            // Recibe los mensajes:
            for (i = 0; i < 5; i++) {
               m = (TextMessage) receptor.receive(1000);
               System.out.println("Mensaje recibido:" + m.getText());
            }
         }
      } catch (JMSException ex) {
         mc.cerrarConexion();
      }
   }
}
```

Podemos probar el ejercicio lanzando *Glasshfish* y creando desde su entorno de administración una factoría de conexiones (*QueueConnectionFactory*) de tipo javax.jms. QueueConnectionFactory y una cola de mensajes (*miCola*) de tipo javax.jms.Queue. Finalmente ya podemos ejecutar la clase *Productor* y después *Consumidor* para probar la práctica.

Un ejemplo de prueba seria el siguiente:



Ejercicio 2

En el segundo ejercicio realizaremos la práctica anterior en un entorno distribuido. La cola de mensajes la ubicaremos en el nodo del Consumidor, de modo que el productor necesitara localizarla.

En primer lugar nos pondremos **del lado del productor**. Crearemos un nuevo proyecto llamado ProyectoJMS2 y la clase MsgConexion como antes, pero ahora cambiaremos el nombre de la factoria de colas por “jms/facotria” y el de la cola por “cola”.

```
public String nombreCola = "cola";
factoria = (QueueConnectionFactory) contexto.lookup("jms/factoria");
```

Crearemos la clase Productor igual que en el ejercicio anterior pero cambiaremos el mensaje de “Hola Mundo” por “Hola Mundo Distante”.

```
m.setText("Hola Mundo distante" + i);     // Contenido mensaje
```

Una vez hecho esto, lanzaremos el servidor glassfish y crear una factoria de conexión de colas llamada “jms/factoria” y ahora incluimos la propiedad addresslist con valor mq://IP_CONSUMIDOR:7676/jms, donde IP_CONSUMIDOR será la ip del consumidor. Crearemos la cola local “cola” de tipo javax.jms.Queue.

Ahora nos pondremos **del lado del consumidor**. Crearemos el proyecto ProyectoJMS2consumidor y añadiremos la clase MsgConexion como antes pero cambiando el nombre de la factoría de colas por “jms/factoria” y el de la cola por “cola”. Crearemos la clase consumidor como en el ejercicio anterior y modificaremos el parámetro del método receive a 0 para que espere de forma indefinida.

```
m = (TextMessage) receptor.receive(0);
```

Probamos el ejercicio ejecutando Productor y Consumidor en sus respectivos nodos.

Podemos comprobar que el consumidor nos muestra los mensajes como en la imagen de abajo.

```
INFORMACION: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version:  5.1  (Build 9-b) Compile:  July 29 2014 1229
oct 23, 2015 9:49:58 PM com.sun.messaging.jms.ra.ResourceAdapter start
INFORMACIÓN: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP
oct 23, 2015 9:49:58 PM com.sun.messaging.jms.ra.ResourceAdapter start
INFORMACIÓN: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE
Mensaje recibido:Hola Mundo distante 0
Mensaje recibido:Hola Mundo distante 1
Mensaje recibido:Hola Mundo distante 2
Mensaje recibido:Hola Mundo distante 3
Mensaje recibido:Hola Mundo distante 4
oct 23, 2015 9:49:59 PM com.sun.messaging.jms.ra.ResourceAdapter stop
INFORMACIÓN: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter stopping...
oct 23, 2015 9:49:59 PM com.sun.messaging.jms.ra.ResourceAdapter stop
INFORMACIÓN: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter stopped.
oct 23, 2015 9:49:59 PM com.sun.enterprise.connectors.service.ResourceAdapterAdminServiceImpl sendStopToResourceAdapter
INFORMACIÓN: RAR7094: jmsra shutdown successful.
run-single:
```