

Ejercicio de MDB

El objetivo es simular un ambiente de ejecución distribuido donde una aplicación instalada en una “máquina” interopera con otra aplicación que reside en el Servidor de Aplicaciones a través de un sistema de mensajería implementado con JMS.

El sistema de mensajería es administrado por un “servidor de aplicaciones”, en nuestro caso Glassfish.

En él se configuran los “destinations” que serán accedidos por las aplicaciones Java para enviar y recibir mensajes.

Se construirán 2 aplicaciones:

- **ProductorMensajes:** contiene una clase Java que crea un mensaje y lo envía a una destination creada en el Glassfish.
- **Modulo EJB:** contiene un MDB que recibe la información de un mensaje y SessionBeans sin Estado que procesa la información.

Configuración de Destination y ConnectionFactory en Glassfish

1- Subir el servidor desde la consola mediante el siguiente comando

asadmin start-domain

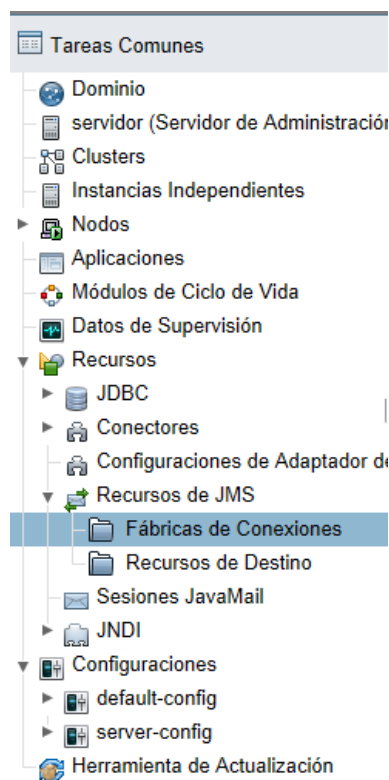
2- Ingresar a la consola administrativa desde un navegador con la siguiente URL:

http://localhost:4848/

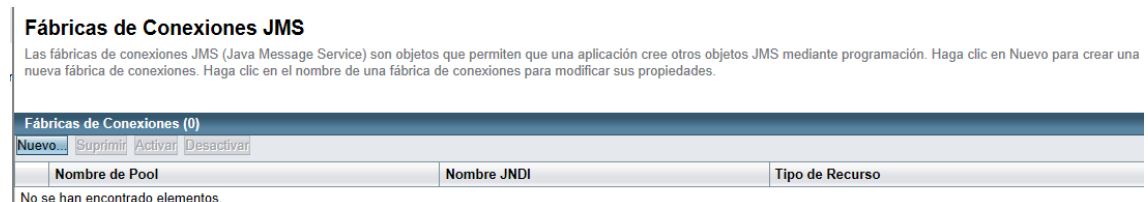
3- Creación del ConnectionFactory

Dentro de las opciones de las tareas comunes

Recursos->Recursos JMS->Fabrica de Conexiones



3.1-Seleccionamos Nuevo



3.2-Ingresamos en el Formulario de Fabrica de Conexiones

Nombre de Pool: *ConnectionFactory*

Tipo de Recurso: *javax.jms.ConnectionFactory*

Nueva Fábrica de Conexiones JMS

[Aceptar](#) [Cancelar](#)

Al crear una nueva fábrica de conexiones JMS (Java Message Service), también se crean un pool de conexiones del conector para la fábrica y un recurso del conector.

Configuración General

Nombre de Pool: *

Tipo de Recurso: *

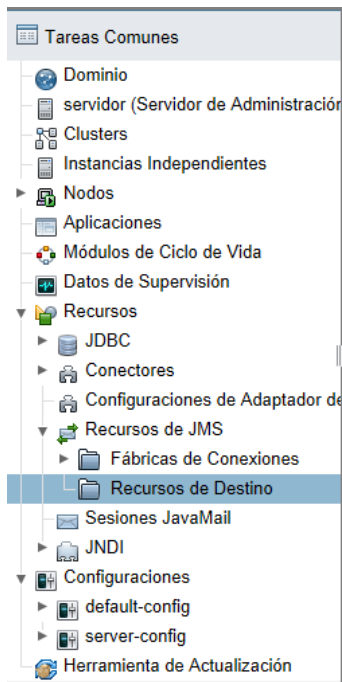
Descripción:

Estado: ☒ Activada

4- Creación del Destination

Dentro de las opciones de las tareas comunes

Recursos->Recursos JMS->Recursos de Destino



4.1-Seleccionamos Nuevo

Recursos de Destino JMS

Los destinos JMS actúan como repositorios para los mensajes. Haga clic en Nuevo para crear un nuevo recurso de destino. Haga clic en el nombre de un recurso de destino para modificar sus propiedades.

Recursos de Destino (0)

[Nuevo...](#) [Suprimir](#) [Activar](#) [Desactivar](#)

Nombre JNDI	Estado	Tipo de Recurso	Descripción
-------------	--------	-----------------	-------------

No se han encontrado elementos.

4.2-Ingresamos en el Formulario de Fabrica de Conexiones

Nombre JNDI: *jms/Queue*

Nombre de Destino Físico: *MiQueue*

Tipo de Recurso: *javax.jms.Queue*

Nuevo Recurso de Destino JMS

[Aceptar](#) [Cancelar](#)

Al crear un nuevo recurso de destino JMS (Java Message Service), también se crea un recurso de objeto de administración.

Nombre JNDI: *

Nombre único de hasta 255 caracteres; sólo debe contener caracteres alfanuméricos, de subrayado, guiones o puntos

Nombre de Destino Físico *

Nombre de destino en el broker de Message Queue. Si el destino no existe, se creará automáticamente cuando sea necesario.

Tipo de Recurso: *

Descripción:

Estado:

☒ Activada

Creación del proyecto Java para la Aplicación Productora de Mensajes

1-En la IDE de NetBeans, crear un proyecto seleccionando de la categoría *Java* la opción *Java Application*.

2- Asignarle el nombre *ProductorMensajes* y definir la ubicación del proyecto en la carpeta *C:\EjercicioJEE*

3-Asegurarse de DESMARCAR las opciones *Set as Main Project* y *Create Main Class* y finalizar.

4-Dentro del proyecto creado, posicionarse en la carpeta *Source Packages* y crear un nuevo *Java Package* con el nombre *uy.edu.ort.productor.mensajes*

5-Agregar las siguientes librerías

gf-client.jar (Ubicada en *C:\Program Files\glassfish-3.1.2.2\glassfish\lib*)

jms.jar (Ubicada en *C:\Program Files\glassfish-3.1.2.2\mq\lib*)

6-Crear la clase Java con nombre *ProductorMensajes* en el *app.productor.mensajes*

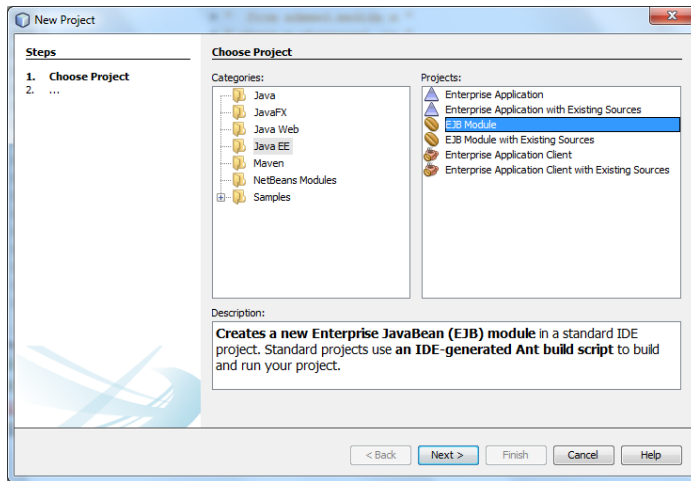
7-Código de la clase *ProductorMensajes*

```
public class ProductorMensajes {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        try {  
            //Seteo las Properties para el contexto  
            Properties props = new Properties();  
            props.setProperty("java.naming.factory.initial",  
"com.sun.enterprise.naming.SerialInitContextFactory");  
            props.setProperty("java.naming.factory.url.pkgs", "com.sun.enterprise.naming");  
            props.setProperty("java.naming.factory.state",  
"com.sun.corba.iiop.impl.presentation.rmi.JNDIStateFactoryImpl");  
            props.setProperty("org.omg.CORBA.ORBInitialHost", "localhost");  
            props.setProperty("org.omg.CORBA.ORBInitialPort", "3700");  
            //Creo el Contexto para obtener los recursos del servidor  
            InitialContext ic = new InitialContext(props);  
  
            // Obtenemos a traves del servicio JNDI la ConnectionFactory del  
            // servidor de aplicaciones  
            ConnectionFactory connectionFactory = (ConnectionFactory)ic.lookup("ConnectionFactory");  
            // Obtenemos a traves del servicio JNDI la "destination" que vamos  
            // a utilizar, en este caso una Queue  
            Queue queue = (Queue) ic.lookup("jms/Queue");  
  
            //Creo la Connection mediante la ConnectionFactory  
            Connection connection= connectionFactory.createConnection();  
            //Creo la Session mediante la Connection  
            Session session= connection.createSession(false, Session.AUTO_ACKNOWLEDGE);  
            //Creo la MessageProducer mediante la Session  
            MessageProducer messageProducer= session.createProducer(queue);  
            //Creo la TextMessage mediante la Session  
            TextMessage textMessage=session.createTextMessage();  
            textMessage.setText("Enviando Mensaje");  
            //Envío el mensaje mediante MessageProducer  
            messageProducer.send(textMessage);  
        }  
    }  
}
```

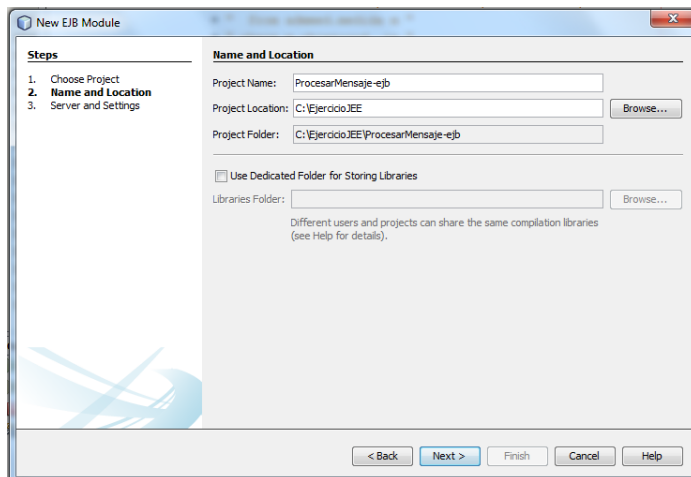
```
    } catch (Exception ex) {  
        Logger.getLogger(ProductorMensajes.class.getName()).log(Level.SEVERE, null, ex);  
    }  
}  
}
```

Creación del proyecto Modulo EJB

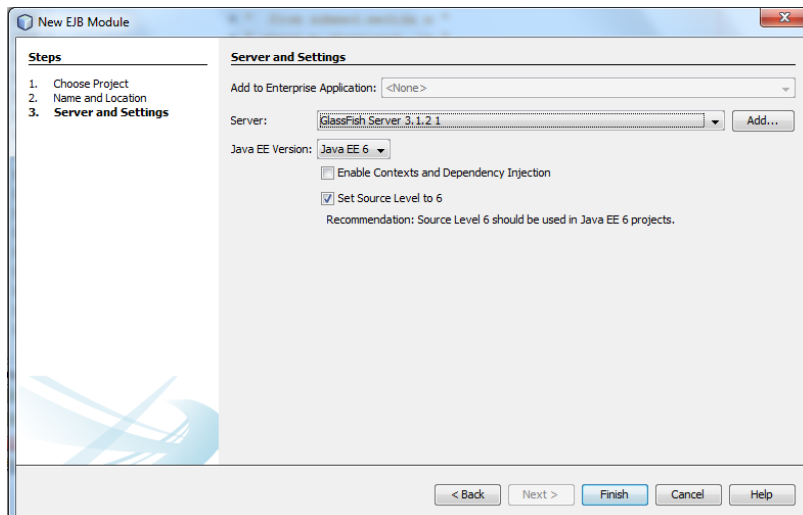
1-En la IDE de NetBeans, crear un proyecto seleccionando de la categoría *Java EE* la opción *Modulo EJB*.



2- Asignarle el nombre *ProcesarMensaje-ejb* y definir la ubicación del proyecto en la carpeta *C:\EjercicioJEE*



3-Seleccionar el *ServidorGlassfish* y la *JEE Version 6*



4-Dentro del proyecto creado, posicionarse en la carpeta Source Packages y crear dos Java Package con los nombre *uy.edu.ort.recibir.mensaje* y *uy.edu.ort.procesar.mensaje*

A- Creación Session Beans

1-Crear un Session Bean con el nombre *ProcesarMensajeSB* en el paquete *uy.edu.ort.procesar.mensaje* de tipo Stateless e interface Local

2-Código del Session Bean

@Local

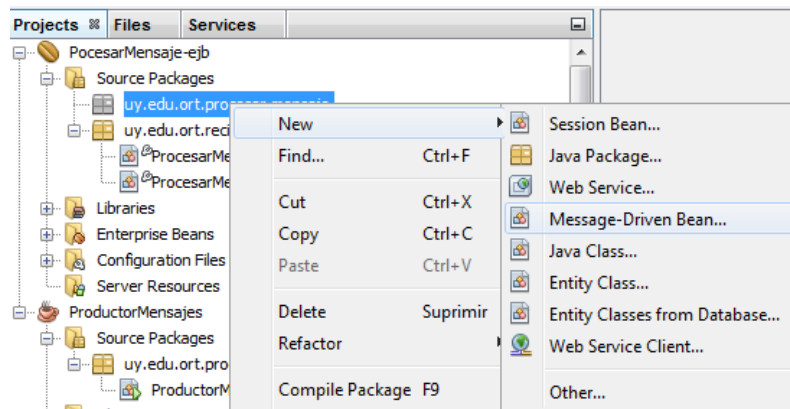
```
public interface ProcesarMensajeSBLocal {  
    public void procesoMensaje(String mensaje);  
}
```

@Stateless

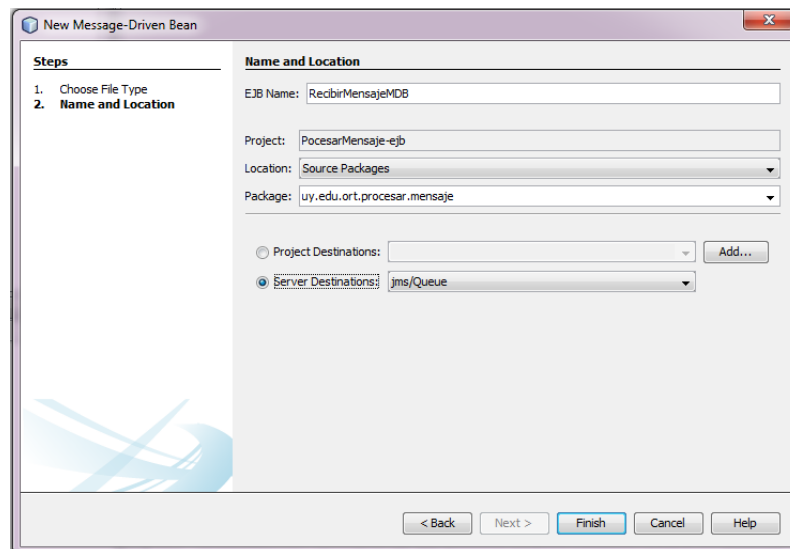
```
public class ProcesarMensajeSB implements ProcesarMensajeSBLocal {  
    public void procesoMensaje(String mensaje){  
        System.out.println(mensaje);  
    }  
}
```


B- Creación MDB

1-Crear un MDB en el paquete uy.edu.ort.recibir.mensaje



2-Crear un MDB con el nombre RecibirMensajeMDB y asignarle el destination creado(jms/Queue)



3-Código del MDB

```
@MessageDriven(mappedName = "jms/Queue", activationConfig = {
    @ActivationConfigProperty(propertyName = "acknowledgeMode", propertyValue = "Auto-
    acknowledge"),
    @ActivationConfigProperty(propertyName = "destinationType", propertyValue =
    "javax.jms.Queue")
})
public class RecibirMensajeMDB implements MessageListener {

    @EJB
    ProcesarMensajeSBLocal mensajeSBLocal;

    public RecibirMensajeMDB() {
    }

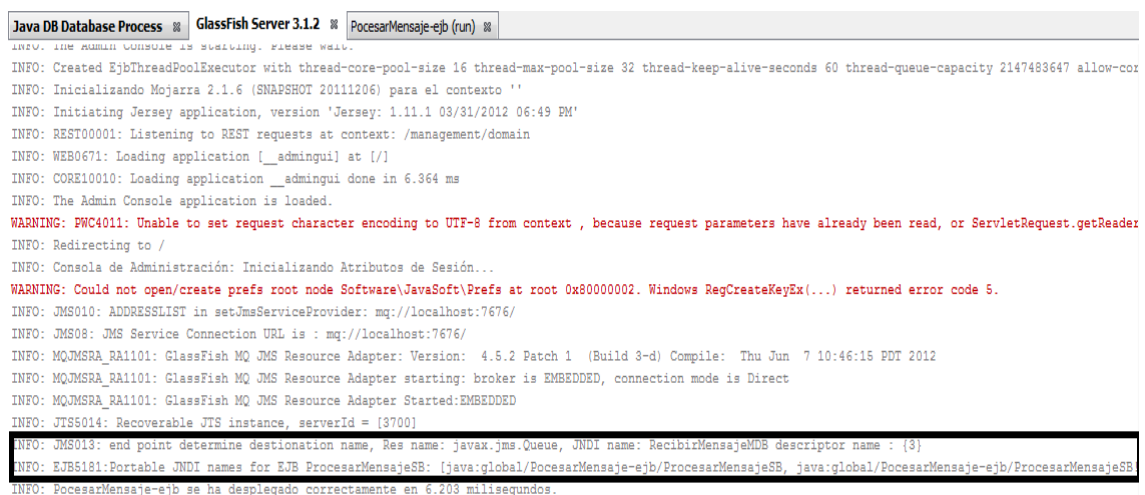
    @Override
    public void onMessage(Message message) {
        try {
            //Verifico el tipo de mensaje
            if (message instanceof TextMessage) {

                TextMessage textMessage = (TextMessage) message;
                //Asigno el procesamiento del mensaje
                mensajeSBLocal.procesarMensaje(textMessage.getText());

            }
        } catch (JMSException ex) {
            Logger.getLogger(RecibirMensajeMDB.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

C-Realizamos el build y Deploy

Verificar que el server esté iniciado. Desde la raíz del proyecto de nuestra aplicación Enterprise seleccionar del menú contextual del mouse la opción Deploy. En la consola de NetBeans podrá verificarse el resultado del deploy:



```
Java DB Database Process  GlassFish Server 3.1.2  PocesarMensaje-ejb (run)
INFO: The Admin Console is starting. Please wait...
INFO: Created EjbThreadPoolExecutor with thread-core-pool-size 16 thread-max-pool-size 32 thread-keep-alive-seconds 60 thread-queue-capacity 2147483647 allow-coor
INFO: Inicializando Mojarra 2.1.6 (SNAPSHOT 20111206) para el contexto ''
INFO: Initiating Jersey application, version 'Jersey: 1.11.1 03/31/2012 06:49 PM'
INFO: REST00001: Listening to REST requests at context: /management/domain
INFO: WEB0671: Loading application [_adminui] at [/]
INFO: CORE10010: Loading application __adminui done in 6.364 ms
INFO: The Admin Console application is loaded.
WARNING: PWC4011: Unable to set request character encoding to UTF-8 from context , because request parameters have already been read, or ServletRequest.getReader
INFO: Redirecting to /
INFO: Consola de Administración: Inicializando Atributos de Sesión...
WARNING: Could not open/create prefs root node Software\JavaSoft\Prefs at root 0x80000002. Windows RegCreateKeyEx(...) returned error code 5.
INFO: JMS010: ADDRESSLIST in setJmsServiceProvider: mq://localhost:7676/
INFO: JMS08: JMS Service Connection URL is : mq://localhost:7676/
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 4.5.2 Patch 1 (Build 3-d) Compile: Thu Jun 7 10:46:15 PDT 2012
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is EMBEDDED, connection mode is Direct
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:EMBEDDED
INFO: JTS5014: Recoverable JTS instance, serverId = {3700}
INFO: JMS013: end point determine destination name, Res name: javax.jms.Queue, JNDI name: RecibirMensajeMDB descriptor name : {3}
INFO: EJB5181:Portable JNDI names for EJB ProcesarMensajeSB: [java:global/PocesarMensaje-ejb/ProcesarMensajeSB, java:global/PocesarMensaje-ejb/ProcesarMensajeSB]
INFO: PocesarMensaje-ejb se ha desplegado correctamente en 6.203 milisegundos.
```