Oracle

Exam 1z0-895

Java EE 6 Enterprise JavaBeans Developer Certified Expert Exam

Version: 14.0

[Total Questions: 90]

Question No : 1

A developer wants to create a JMS message-driven bean that responds to javax.jms.TextMessage messages. Which two statements are true? (Choose two.)

A. The developer must implement the ejbCreate method.
B. The developer does NOT need to create a business interface for the bean.
C. The developer must implement a method that declares javax.jms.TextMessage as an argument.
D. The message-driven bean class must implement methods of the javax.jms.TextMessageListener interface.
E. The message-driven bean class must implement methods of the javax.ejb.MessageDrivenBean interface.

Answer: B,D

Explanation: B: *Client components do not locate message-driven beans and invoke methods directly on them. Instead, a client accesses a message-driven bean through, for example, JMS by sending messages to the message destination for which the message-driven bean class is the MessageListener.

D: *A message-driven bean is an enterprise bean that allows Java EE applications to process messages asynchronously. This type of bean normally acts as a JMS message listener, which is similar to an event listener but receives JMS messages instead of events.

*In a fashion similar to a Message-Driven Bean (MDB) in the EJB world, the Message-Driven POJO (MDP) acts as a receiver for JMS messages. The one restriction (but see also below for the discussion of the MessageListenerAdapter class) on an MDP is that it must implement the javax.jms.MessageListener interface.

Reference:The Java EE 6 Tutorial,What Is a Message-Driven Bean?

Question No : 2

A developer writes a stateful session bean called FooBean.

Which code can be inserted before Line 11 of the FooBean class to define a TYPE-level environment dependency on a JMS Topic?

```
11. public class FooBean {

12.

13.  public void foo() ()

14.

15. }
```

A. @Resource (type=Topic.class)
B. @Resource (name="topicRef")
Private static Topic topic;
C. @Resource private Topic topic
D. @Resource(name="topicRef", type=Topic.class)

Answer: D

Explanation: @Resource can decorate a class, a field or a method at runtime/init through injection. (name, type, authenticationType, shareable, mappedName, description)
Type- level env dependency on a Topic - @Resource(name="topicRef", type=Topic.class)

Question No : 3

A java EE application contains a session bean which uses a security role USER. A group called people is defined an LDAP server. Which two define appropriate EJB role responsibilities? (Choose two.)

A. The deployer defines and configures the LDAP realm.
B. The system administrator defines and configures the LDAP realm.
C. The deployer maps the application role USER to the LDAP group people.
D. The system administrator maps the application role USER to the LDAP group people.

Answer: B,C

Question No : 4

A developer needs to deliver a large-scale enterprise application that connects developer chooses an EJB 3.1-compliant application server, which three are true about the EJB business component tier? (Choose three.)

A. Load-balancing is NOT a guarantee for all EJB 3.1 containers.
B. Clustering is guaranteed to be supported by the EJB 3.1 container.
C. Thread pooling can be optimized by the Bean Provider programmatically.
D. Bean Providers are NOT required to write code for transaction demarcation.
E. Support for server fail-over is guaranteed for an EJB 3.1-compliant application server.
F. EJB 3.1 compliant components are guaranteed to work within any Java EE 6 application server

Answer: A,C,F

Explanation: The EJB tier hosts the business logic of a J2EE application and provides system-level services to the business componentsproblems include state maintenance, transaction management, and availability to local and remote clients.

The EJB 3.1 specification does not address "high-end" features like clustering (not B), load-balancing (A) and fail-over (not E).

F: The target platform for EJB is Java EE.

Question No : 5

A bean developer writes a stateless session bean FooEJB with the following asynchronous business method:

```
@Asynchronous

public Future<Integer> fooAsync () {

System.out.printIn (“begin”);

int i = 1;

System.out.print(“end”);

Return new AsyncResult<Integer> (i);

}
```

Given the following code, where fooRef is an EJB reference to FooEJB:

```
Future<Integer> fooFuture = fooref.fooAsync();
```

fooFuture.cancel (true);

Which two represents possible system output after all processing has completed? (Choose two)

A. Begin end
B. Begin
C. End
D. 1
E. <no output>

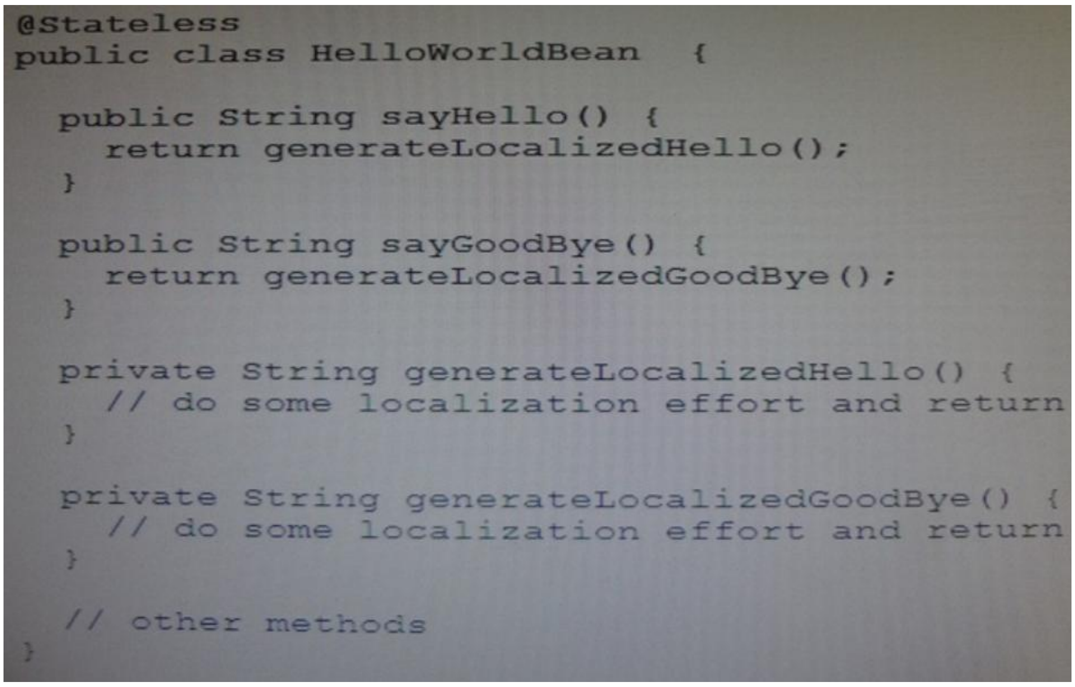
Answer: D,E

Explanation: Either it will run and return 1, or it will be cancelled and produce no output.

Note:EJB 3.1 can support a return type of java.util.concurrent.Future<V>, where V represents the resultant value of an asynchronous invocation. In case you are unfamiliar with it, the Future<V> interface allows you to do things like cancelling an asynchronous invocation, checking if an invocation is complete, check for exceptions and getting the results of an asynchronous invocation.

Question No : 6

Suppose an EJB component is named HelloWorldBean is deployed as a standalone ejb-jar. Assuming the HelloWorldBean is implemented as follows:



```
@Stateless
public class HelloWorldBean  {

  public String sayHello() {
    return generateLocalizedHello();
  }

  public String sayGoodBye()  {
    return generateLocalizedGoodBye();
  }

  private String generateLocalizedHello()  {
    // do some localization effort and return
  }

  private String generateLocalizedGoodBye()  {
    // do some localization effort and return
  }

  // other methods
}
```

Which types of clients are guaranteed to have access to HelloWorldBean:

A. Java EE application client container applications
B. Java EE ejb components within the same ejb-jar
C. Java EE web-tier component applications deployed in the same container
D. Java EE ejb component applications deployed in the same container

Answer: D

Question No : 7

You are writing a client that sends a message to a JMS queue. Which statement is true?

A. You use a connection factory to create a session.
B. When you create a session, you specify whether or not it is transacted.
C. When you create a connection, you specify the acknowledgment mode.
D. When you create a message producer, you must specify the name of the destination to which you will send messages.

Answer: A

Explanation: Note:
The SimpleMessageClient sends messages to the queue that the SimpleMessageBean listens to.
The client starts by injecting the connection factory and queue resources:

```
@Resource(mappedName="jms/ConnectionFactory")
private static ConnectionFactory connectionFactory;
@Resource(mappedName="jms/Queue")
private static Queue queue;
```

Next, the client creates the connection, session, and message producer:

```
connection = connectionFactory.createConnection();
session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
messageProducer = session.createProducer(queue);
```

Finally, the client sends several messages to the queue:

```
message = session.createTextMessage();
for (int i = 0; i < NUM_MSGS; i++) {
message.setText("This is message " + (i + 1));
System.out.println("Sending message: " + message.getText());
messageProducer.send(message);
}
```

Question No : 8

How many interceptor classes can be applied to a single stateful session bean?

A. a maximum of one
B. any number may be applied
C. one for each business method
D. one for each business interface

Answer: B

Explanation: The @Interceptors annotation can take an array of classes, so you can bind more than one class-level interceptor this way, e.g.

```
@Stateless
@Interceptors ({TracingInterceptor.class, SomeInterceptor.class})
public class EmailSystemBean
{
```

}

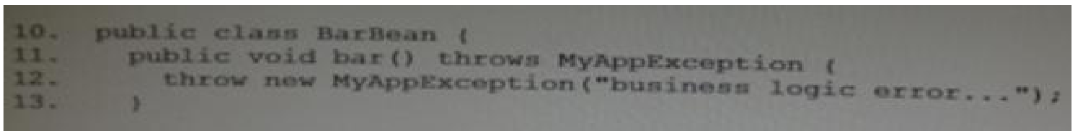
Reference: EJB Interceptors

http://docs.jboss.org/ejb3/app-server/tutorial/interceptor/interceptor.html

Question No : 9

FooBean and BarBean are both EJB 3.x stateless session beans with bean-managed transaction demarcation. The business method foo in FooBean starts a UserTransaction and invokes the business method bar in BrBean.

Given:



```
10.   public class BarBean {
11.     public void bar() throws MyAppException {
12.       throw new MyAppException("business logic error...");
13.     }
```

What is the expected result of this method invocation assuming control reaches Line 12?

A. FooBean.foo method receives MyAppException.
B. The container discards the BarBean bean instance.
C. FooBean.foo method receives a javax.ejb.EJBException that wraps MyAppException.
D. FooBean.foo method receives javax.transaction.TransactionRolledbackException.

Answer: D
Explanation: The transaction will roll back.

Note:
*In bean-managed transaction demarcation, the code in the session or message-driven bean explicitly marks the boundaries of the transaction. Although beans with container-managed transactions require less coding, they have one limitation: When a method is executing, it can be associated with either a single transaction or no transaction at all. If this limitation will make coding your bean difficult, you should consider using bean-managed transactions.

Reference:The Java EE 5 Tutorial,Bean-Managed Transactions

Question No : 10

Which statement is true about both stateful session beans and stateless session beans?

A. Bean instance are NOT required to survive container crashes.
B. Any bean instance must be able to handle concurrent invocations from different threads.
C. A bean with bean-managed transactions must commit or roll back any transaction before returning from a business method.
D. The container passivates and actives them using methods annotated with @PrePassivate and @PostActivate annotations.

Answer: A,C

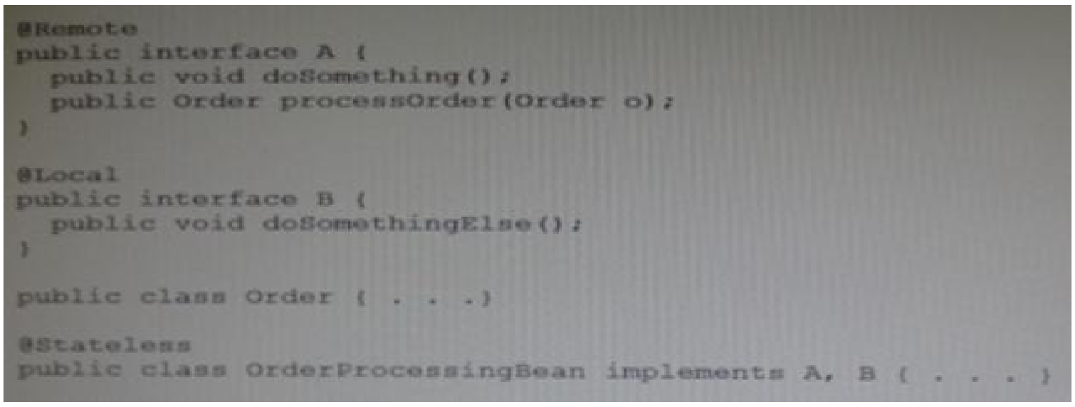
Explanation:

Note:
*Session beans can either be stateful or stateless. With stateful beans, the EJB container saves internal bean data during and in between method calls on the client's behalf. With stateless beans, the clients may call any available instance of an instantiated bean for as long as the EJB container has the ability to pool stateless beans. This enables the number of instantiations of a bean to be reduced, thereby reducing required resources.

Incorrect:
B:Stateful session beans maintain state both within and between transactions. Each stateful session bean is therefore associated with a specific client.
D:@PrePassivate(javax.ejb.PrePassivate) :
If a stateful session bean instance is idle for too long, the container might passivate it and store its state to a cache.
The method tagged by this annotation is called before the container passivates the bean instance.
This annotation is only applicable to stateful session beans.

Question No : 11

You have been tasked to build a jar file that can be used by a Java SE client to access the remote instance of the OrderProcessingBean. Given the following design:



```
@Remote
public interface A {
   public void doSomething();
   public Order processOrder(Order o);
}

@Local
public interface B {
   public void doSomethingElse();
}

public class Order { . . .}

@Stateless
public class OrderProcessingBean implements A, B { . . . }
```

Which classes would need to be included in the client jar file?

A. B, Order
B. A, Order
C. A, B, Order
D. A, B, Order, OrderProcessingBean

Answer: B

Explanation:

Note:

*An EJB client JAR file is an optional JAR file that can contain all the class files that a client program needs to use the client view of the enterprise beans that are contained in the EJB JAR file.

*If all your EJBs are in the same EAR then you can use local interfaces, if not you need remote interfaces.

Question No : 12

Given two stateless session beans, ABean and BBean: