Testking.1z0-895.90.QA

Number: 1z0-895
Passing Score: 800
Time Limit: 120 min
File Version: 29.0

Testking

1z0-895

Java EE 6 Enterprise JavaBeans Developer Certified Expert Exam

Exam A

QUESTION 1
A developer wants to write a stateful session bean using the following interface as local business interface:

```
1. package acme;
2. public interface Bar {
3. public void bar ();
4. }
```

Assuming there is not an ejb-jar.xml file, which code can be inserted into Lines 4-6 below to define the bean with the ejb name of BarBean?

```
1. package acme;
2. import javax.ejb.*;
3. import java.io.*;
4.
5.
6.
7. }
```

A. @Statefulpublic class BarEJB implements Bar {public void bar () {}
B. @Stateful (name = "Bar")public class Barbean implements Bar {public void bar () {}
C. @Statefulpublic class BarBean implements Serializable, Bar { public void bar () {}
D. @Stateful (name = "bar")public class BarBean implements Serializable, Bar {public void bar () throws java.rmi.RemoteException {}

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 2
A developer creates a stateful session bean that is used by many concurrent clients. The clients are written by other development team; and it is assumed that these clients might not remove the bean when ending their session. The number of concurrent sessions will be greater than the defined bean cache size.

The developer must consider that the state of the session bean can be influenced by either passivation or timeout.

Which three actions should the developer take to make the bean behave correctly in passivation and timeout situations? (Choose three.)

A. Release references to resources in a @Remove annotated method.
B. Re-establish references to resources in an @Init annotated method.
C. Release references to resources in a @PreDestroy annotated method.
D. Release references to resources in a @PrePassivate annotated method.
E. Re-establish references to resources in a @PostActivate annotated method.

Correct Answer: CDE
Section: (none)
Explanation

Explanation/Reference:

QUESTION 3
A stateful session bean contains a number of instance variables. The types of instance variables A and B are serializable. Instance variable B is a complex type which is populated by many business calls, and can, therefore, not be refilled by the client without starting all over. A helper instance variable C is defined as having a Serializable type, and can hold all the information which is in variable B. for example, B is of type XML-DOM tree and C of Type String.

Which two solutions, when combined, maintain the state of the session bean over a passivation and activation by the container? (Choose two.)

A. The value of helper variable C is used to create the value of Instance variable B in the beans no-arg constructor.
B. The value of helper variable C is used to create the value of instance variable B in a @postcreate annotated method.
C. The value of helper variable C is used to create the value of instance variable B in a @postActivate annotated method.
D. Instance variable A must be made null and instance variable B must be converted to a Serializable type and assigned to another instance variable in a @preDestroy annotated method.
E. Instance variable A must be defined transient. Instance variable B must be converted to a Serializable type, set to null, and assigned to the instance variable C in a @PrePassivate annotated method.

Correct Answer: CE
Section: (none)
Explanation

Explanation/Reference:

QUESTION 4
A developer writes a stateful session bean FooBean with one remote business interface Foo. Foo defines an integer / setter method pair implemented as:

```
10. private int value;
11. public void setValue (int i) {value = i; }
12. public int getValue () {return value; }
```

A session bean ClientBean has a business method doSomething and an ejb-ref with ejb-ref-name "fooRef" that is mapped to FooBean's Foo interface.

```
11. @Resource private SessionContext SessionCtx;
12. public void doSomething () {
13. Foo foo1 = (Foo) sessionCtx.lookup("fooRef");
14. Foo foo2 = (Foo) sessionCtx.lookup("fooRef");
15. foo1.setvalue(1);
```

Which statement is true after the code at line 15 completes?

A. Foo1.getValue () = = 0 and foo2.getValue() = = 0
B. Foo1.getValue () = = 0 and foo2.getValue() = = 1
C. Foo1.getValue () = = 1 and foo2.getValue() = = 0
D. Foo1.getValue () = = 1 and foo2.getValue() = = 1

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

Still Valid.

QUESTION 5
A developer writes a stateless session bean FooBean with one remote business interface FooRemote containing one business method foo. Method foo takes a single parameter of application-defined type MyData.

```
11. public class MyData implements java.io.Serialization {
12. int a;
13. }
```

Methods foo is implemented with the FooBean class as:

```
11. public void foo (MyData data) {
12. data.a = 2;
13. }
```

Another session bean within the same application has a reference to FooRemote in variable fooRef and calls method foo with the following code:

```
11. MyData data = new MyData();
12. data.a = 1;
13. fooRef.foo(data);
14. System.out.println(data.a);
```

What is the value of data.a when control reaches Line 14 of the client?

A. 0
B. 1
C. 2

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

QUESTION 6
Assume a client will be accessing a Singleton bean.

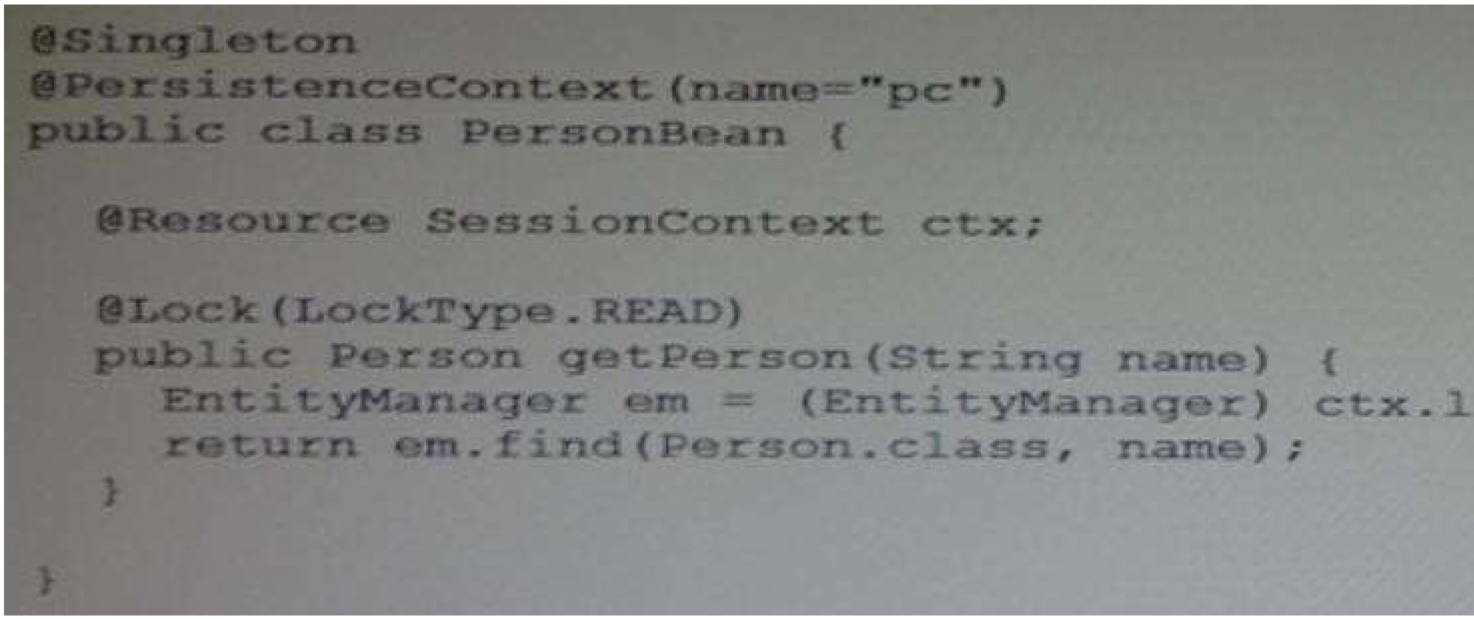
Which client views is a Singleton bean capable of exposing? (Choose two)

A. Web Service
B. Message listener
C. EJB 2.x Remote Home
D. EJB 3.x local business
E. Java Persistence API entity

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation:
JSR 318 Enterprise JavaBeans 3.1, Final Release: 4.8.6Operations Allowed in the Methods of a Singleton Session BeanInvoking the getEJBObject and getEJBHome methods is disallowed since a singleton session bean does not support the EJB 2.x Remote client view.·Invoking the getEJBLocalObject and getEJBLocalHome methods is disallowed since a singleton session bean does not support the EJB 2.x Local client view.

QUESTION 7
A developer writes a Singleton bean that uses the java Persistence API within a business method:



```
@Singleton
@PersistenceContext(name="pc")
public class PersonBean {

  @Resource SessionContext ctx;

  @Lock(LockType.READ)
  public Person getPerson(String name) {
    EntityManager em = (EntityManager) ctx.l
    return em.find(Person.class, name);
  }

}
```

Two different concurrently executing caller threads acquire an EJB reference to PersonBean and each invoke the getPerson () method one time. How many distinct transaction are used to process the caller invocations?

A. 0
B. 1
C. 2

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Only one transaction is required. LockType READ allows simultaneous access to singleton beans.

Note: READ

public static final LockType READ
For read-only operations. Allows simultaneous access to methods designated as READ, as long as no WRITE lock is held.

QUESTION 8
Given the following client-side code that makes use of the session bean Foo:

10. @EJB Foo bean1;
12. @EJB Foo bean2;

// more code here

20. boolean test1 = beanl.equals(bean1);
21. boolean test2 = beanl.equals(bean2) ;

Which three statements are true? (Choose three)

A. If Foo is stateful, test1 is true, and test2 is true.
B. If Foo is stateful, test1 is true, and test2 is false.
C. If Foo is stateless, test1 is true, and test2 is true.
D. If Foo is stateless, test1 is true, and test2 is false.
E. If Foo is singleton, test1 is true, and test2 is true.
F. If Foo is singleton, test1 is true, and test2 is false.

Correct Answer: BCE
Section: (none)
Explanation

Explanation/Reference:

QUESTION 9
A developer writes a stateful session bean FooBean with two local business interfaces Foo and bar. The developer wants to write a business method called getBar for interface Foo that returns a Bar reference to the same session bean identity on which the client onvokes getBar. Which code, when inserted on line 12 below implements the getBar method with the wanted behavior?

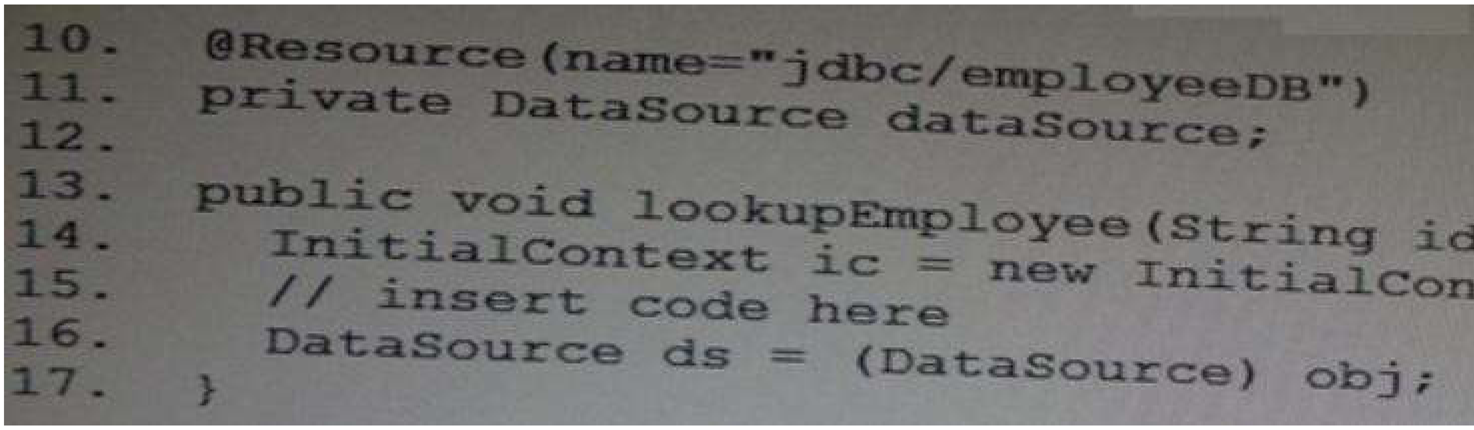
```
10. @Resource SessionContext sessionCtx;
11. public Bar getbar () {
12.
13. }
```

A. Return (bar) this;
B. Return (bar) new FooBarBean();
C. Return (bar) sessionCtx.lookup("FooBarBean")
D. Return (bar) sessionCtx.getBusinessObject(Bar.class);
E. Return (bar) session Ctx.lookup("java: comp/env/ejb/FooBarBean");

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

QUESTION 10
Given the following code in an EJB session bean:



```
10.   @Resource(name="jdbc/employeeDB")
11.   private DataSource dataSource;
12.
13.   public void lookupEmployee(String id
14.      InitialContext ic = new InitialCon
15.      // insert code here
16.      DataSource ds = (DataSource) obj;
17.   }
```

Which code, inserted at Line 15, portably looks up the injected resource?

A. Object obj = ic.lookup ("employeeDB");
B. Object obj = ic.lookup ("dataSource");
C. Object obj = ic.lookup ("jdbc/employeeDB");
D. Object obj = ic.lookup ("java:comp/env/employeeDB");
E. Object obj = ic.lookup ("java:cmp/env/jdbc/employeeDB);

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 11
A developer is writing client code to access a session bean deployed to a server instance. The client can access the session bean under which of these circumstances? (Choose three)

A. The client is deployed in the same JVM as the session bean and the session bean has a local interface.
B. The client is deployed in the same JVM as the session bean and the session bean exposes a no- interface view.
C. The client is deployed in a different JVM from the session bean and the session bean has a local interface.
D. The client is deployed in a different JVM from the session bean and the session bean has a remote interface.
E. The client is deployed in a different JVM from the session bean and the session bean has a no- interface implementation.

Correct Answer: ABD
Section: (none)
Explanation

Explanation/Reference:
Explanation:
D: If your architecture has a requirement whereby the client application (web application or rich client) has to run on a different JavaVirtual Machine (JVM) from the one that is used to run the session beans in an EJB container, then you need to use the remote interface.

QUESTION 12
A developer writes three interceptor classes: AInt, BInt, and CInt. Each interceptor class defines an AroundInvoke method called interceptor. In the ejb-jar.xml descriptor, CInt is declared as the default interceptor.

FooBean is a stateless session bean with a local business interface Foo that declares a method Foo ():

```
10. @Stateless
11. @Interceptors(AInt.class)
12. public class FooBean Implements Foo {
13.
14. @Interceptors (BInt.class)
15. @ExcludeClassInterceptors
16. public void foo () {}
17. }
```

What is the interceptor order when the business method foo () is invoked?

A. BInt
B. CInt, BInt
C. CInt, AInt, BInt
D. BInt, AInt, CInt

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The default Intercepter, CInt, comes first. The class intercepter AInt is excluded by @ExcludeClassInterceptors, so the Method Intercepter BInt would be next in order.

Note 1: By default the ordering of interceptors when invoking a method are

* External interceptors
** Default interceptors, if present
** Class interceptors, if present
** Method interceptors, if present
* Bean class interceptor method
Note 2: Annotation Type ExcludeClassInterceptors
Used to exclude class-level interceptors for a business method or timeout method of a target class.

QUESTION 13
An ejb-jar also contains three interceptor classes: AInt, BInt, CInt. Each interceptor class defines an AroundInvoke method called intercept.

The ejb-jar also contains a stateless session bean FooBean with a local business interface Foo that declares a method foo ():

```
10. @Stateless
11. @Intercaptors ({CInt.class, BInt.class})
12. public class FooBean implements Foo {
13.
14. public void foo () {}
15.
16. }
```

The ejb-jar contains a META-INF/ejb-jar.xml file with an <interceptor-binding> section:

```
<interceptor-binding>
<ejb-name>FooBean</ejb-name>
<interceptor-order>
<interceptor.class>com.acme.AInt</interceptor-class>
</interceptor-order>
</interceptor.binding>
```

What is the interceptor order when the business methodfoo() is invoked?

A. AInt
B. AInt, CInt, BInt
C. CInt, BInt, AInt
D. AInt, BInt, CInt

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
With the interceptor-order clauses Aint will be first in the order of interceptors. Within each group (default, class, method) the order of the interceptors are from left to right as defined in the @Interceptors annotation, and then the xml interceptors. In this scenario, with the @Intercaptors ({CInt.class, BInt.class}) line, the ordering continues with CInt and BInt.

Note 1: By default the ordering of interceptors when invoking a method are

* External interceptors
** Default interceptors, if present
** Class interceptors, if present
** Method interceptors, if present
* Bean class interceptor method
Note 2: You can override the default sort order of the external interceptors by specifiying an interceptor-binding with an interceptor-order specifying the order of the interceptors

QUESTION 14
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

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
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
{
}
```

QUESTION 15
A developer writes an interceptor class called FooInterceptor containing the following AroundInvoke method:

```
11. @AroundInvoke
12. public Object intercept (InvocationContext ctx) {
13. return "intercepted";
14. }
```

FooInterceptor is applied to a business method in a stateless session bean:

```
11. @Interceptors (FooInterceptor.class)
12. public String testzero(int i) {
13. return (i = = 0) ? "zero": "not zero"
14. }
```

Which describes the result when a client invokes the testzero method with a value of 1?

A. The interceptor method is NEVER invoked.
B. The client receives a return value of "zero".
C. The client receives a return value of "not zero".
D. The client receives a return value of "intercepted".

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Updated.

QUESTION 16
A bean developer wants to write a stateless session bean class that implements the following remote business interface:

@Remote
Public interface Foo {
Void bar () throws Exception;

Which bean class method is valid?

A. @Asynchronouspublic void bar () throws Exception { . . . }
B. @AsynchronousFuture <void> bar () { . . .}
C. void bar () throws Exception { . . . }
D. public void bar () { . . . }

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

QUESTION 17
An enterprise bean has security permissions set up using declarative security features. Under which two conditions can a client be guaranteed to have permission to invoke a business method on the enterprise bean? (Choose two.)

A. The Application Assembler has marked the enterprise bean method as unchecked.
B. The client's principal has been assigned a security role with permission to invoke the method.
C. The Application Assembler has set the security-identity deployment descriptor to run-as.
D. The Application Assembler has mapped all security role references using the role-link element.

Correct Answer: AB
Section: (none)
Explanation

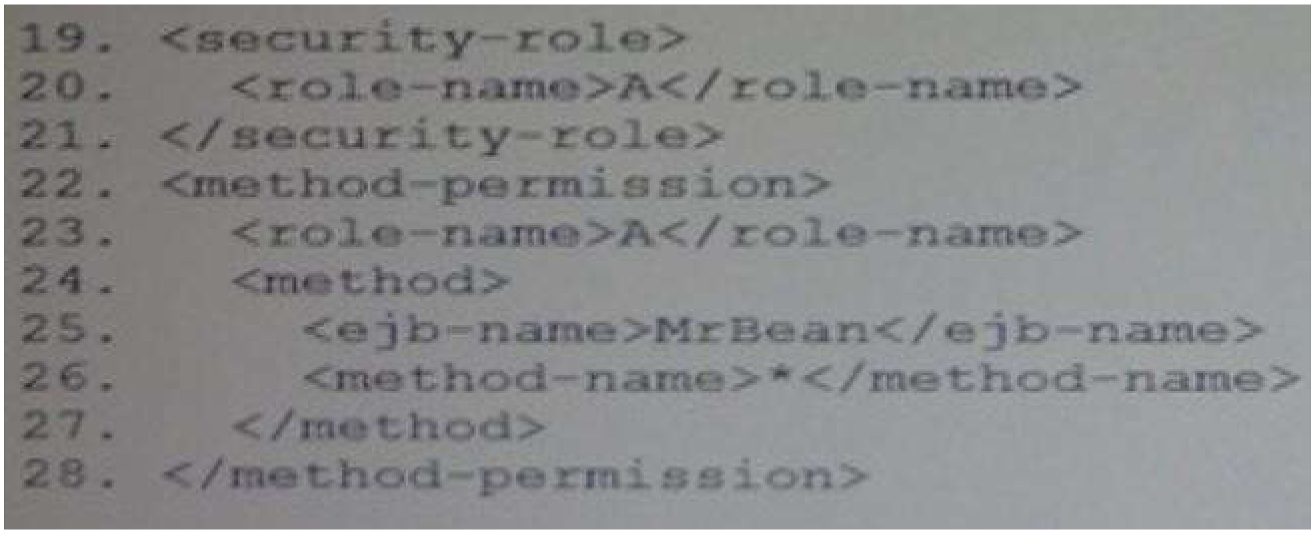
Explanation/Reference:

QUESTION 18
Given a session bean defines:

11. @Stateless (name = "MrBean")
12. public class source SecureBean implements local business, remoteBusiness {

Where LocalBusiness is a local business interface and RemoteBusines is a remote business interface.

The following deployment descriptor entries represent the only security-related metadata:



```
<security-role>
   <role-name>A</role-name>
</security-role>
<method-permission>
   <role-name>A</role-name>
   <method>
      <ejb-name>MrBean</ejb-name>
      <method-name>*</method-name>
   </method>
</method-permission>
```

Which is true about the security roles of clients MrBean?

A. Only LocalBusiness clients must be in role A.
B. Only LocalBusiness clients must be in role B.
C. Both LocalBusiness and RemoteBusiness Clients must be in role A.
D. Both LocalBusiness and RemoteBusiness clients must NOT be in role A.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 19
Suppose a developer wants to create an automatic persistent timer that performs data validation every hour.
Given the following stateless session bean:

@Stateless
Public class OrderVerificationBean {

Private void verificationExternalOrders () {
/ / do something
}
}

What is the minimum modification you would need to make to the bean to create the automatic persistent timer?

A. Modify the verifyExternalOrders methos to look like this: @Scheduleprivate void verifyExternalOrders () {/ do something}
B. Modify the verifyExternalOrders method to look like this: @Schedule (hour = "*")private void verifyExternalOrders () {/ / do something}
C. Modify the verifyExternalOrders method to look like this: @Schedule (persistent = true)private void

verifyExceptionalOrders () {/ / do something}

D. Modify the verifyExternalOrders method to look like this: @Schedule (hour = "*", persistent = true)private void verifyExceptionalOrders () {/ / do something}

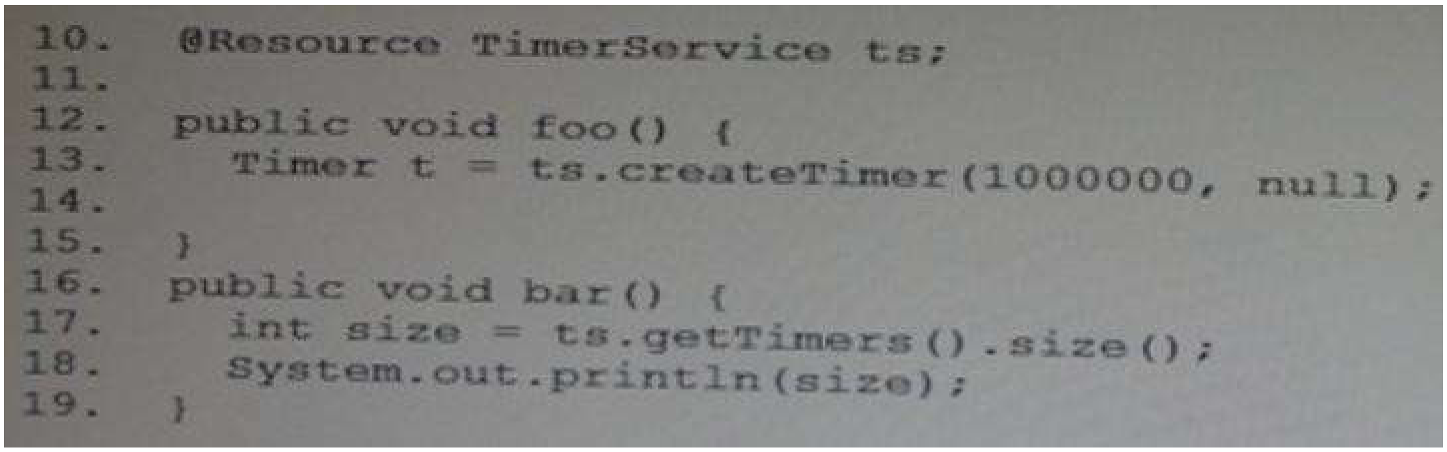
Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

Not D: Timers are persistent by default. If the server is shut down or crashes, persistent timers are saved and will become active again when the server is restarted. If a persistent timer expires while the server is down, the container will call the @Timeout method when the server is restarted.

Nonpersistent programmatic timers are created by calling TimerConfig.setPersistent(false) and passing the TimerConfig object to one of the timer-creation methods.

QUESTION 20
A developer implements a stateless session bean as a timed object. The bean contains two local business methods with the transaction attribute REQUIRED.



```
10.  @Resource TimerService ts;
11.
12.  public void foo() {
13.    Timer t = ts.createTimer(1000000, null);
14.
15.  }
16.  public void bar() {
17.    int size = ts.getTimers().size();
18.    System.out.println(size);
19.  }
```

A client begins a UserTransaction and calls the foo local business method. The foo method returns five seconds later. The client rolls back the transaction and then calls the bar local business method. Assuming there have been no other client invocations on the stateless session bean, what is the value of the size variable when control reaches Line 18?

A. 0
B. 1
C. -1

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:
An enterprise bean usually creates a timer within a transaction. If this transaction is rolled back, the timer creation also is rolled back.
The transaction will roll back. The creation of the timer will rock back. There will be zero timers.

Note: The UserTransaction interface defines the methods that allow an application to explicitly manage

transaction boundaries.
rollback() rolls back the transaction associated with the current thread.

Note 2: size()
Returns the number of elements in this collection. If this collection contains more than Integer.MAX_VALUE elements, returns Integer.MAX_VALUE.

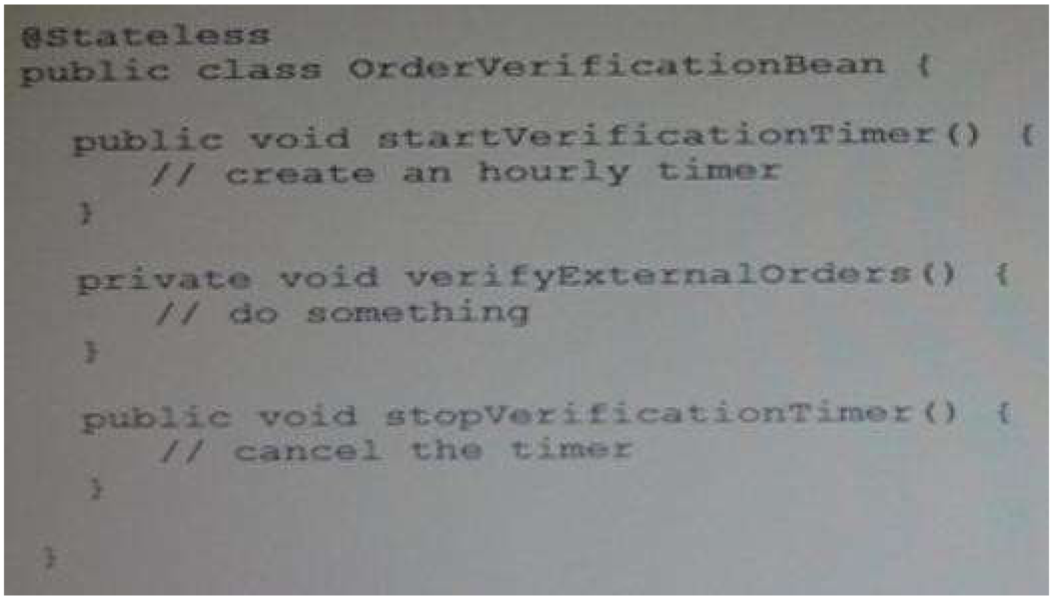
QUESTION 21
A developer wants to create an enterprise bean that uses the EJB Timer service.
Which two are true? (Choose two.)

A. Once created, a timer cannot be canceled.
B. The bean can be used within a transaction.
C. Once created, the timer will survive a container crash.
D. The enterprise bean must implement the TimedObject interface.
E. The developer can use either a message-driven bean, stateless session bean, or stateful session bean.

Correct Answer: BC
Section: (none)
Explanation

Explanation/Reference:
Explanation:
B: An enterprise bean usually creates a timer within a transaction. If this transaction is rolled back, the timer creation also is rolled back. Similarly, if a bean cancels a timer within a transaction that gets rolled back, the timer cancellation is rolled back.
C: Timers are persistent by default. If the server is shut down or crashes, persistent timers are saved and will become active again when the server is restarted. If a persistent timer expires while the server is down, the container will call the @Timeout method when the server is restarted.

QUESTION 22
Suppose developer wants to create an EJB component that performs data validation every hour. Given the following Stateless session bean:


```
@Stateless
public class OrderVerificationBean {

  public void startVerificationTimer() {
     // create an hourly timer
  }

  private void verifyExternalOrders() {
     // do something
  }

  public void stopVerificationTimer() {
     // cancel the timer
  }

}
```

What is the minimum modification you would need to make to the bean to support notification from the TimerService once the timer expires?

A. Modify the verify external orders method to look like this: @TimedOutprivate void verifyExternalOrders () {/ / do something}
B. Modify the verify external orders method to look like this: @EjbTimeOutprivate void verifyExternalOrders () {/ / do something}
C. Modify the verify external orders method to look like this: @ejbTimeOutprivate void verifyExternalOrders () {/ / do something}
D. Modify the verify external orders method to look like this: @TimeOutprivate void verifyExternalOrders () {/ / do something}

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Programmatic Timers
When a programmatic timer expires (goes off), the container calls the method annotated @Timeout in the bean's implementation class. The @Timeout method contains the business logic that handles the timed event.

The @Timeout Method
Methods annotated @Timeout in the enterprise bean class must return void and optionally take a javax.ejb.Timer object as the only parameter. They may not throw application exceptions.

```
@Timeout
public void timeout(Timer timer) {
System.out.println("TimerBean: timeout occurred");
}
```

QUESTION 23
An enterprise developer needs to modify the order of interceptor method execution specified by the Bean Provider, but does NOT have access to the bean's source code. No deployment descriptor was provided in the EJB jar delivered by the Bean Provider.
Which represents the solution to this problem?

A. No solution is possible under these conditions.
B. The Deployer can add metadata annotations to the ejb-jar.
C. The Application Assembler can add metadata annotations to the ejb-jar.
D. The System Administrator can add interceptor binding information at runtime, using vendor-specific tools.
E. The Application Assembler can add a deployment descriptor to the ejb-jar that includes interceptor binding information.

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 24
A developer wants to package an enterprise bean FooBean within a .war file:

```
@Stateless
public xlass FooBean {

public void foo () {}
}
```

Which package approach is correct?

A. / (Root)I META INF /I acme I FooBean.class
B. / (Root)I acme I FooBean.class
C. / (Root)I WEB INF /I acme I FooBean.class
D. / (Root)I WEB INF /I Classes/ I acme I FooBean.class

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
To include enterprise bean class files in aWARmodule, the class files should be in the WEB-INF/classes directory.

Note: Enterprise beans often provide the business logic of a web application. In these cases, packaging the enterprise bean within the web application'sWARmodule simplifies deployment and application organization. Enterprise beans may be packaged within aWARmodule as Java programming language class files or within a JAR file that is bundled within theWARmodule.

QUESTION 25
A developer creates a stateless session bean, EmployeeServiceBean, and its interface, EmployeeService. The session bean uses two annotated entity classes, Employee.class and Department.class.

Which two package options can the developer use when creating a deployable EAR? The proposed directory structure is listed for each option. (Choose two)

A. Emp.earemp-ejb.jarMETA-INF/persistence.xmlEmployeeService.classEmployeeServiceBean.classlib/emp-classes.jarEmployee.classDepartment.class
B. Emp.earMETA-INF/orm.xmlEmployeeService.classEmployeeServiceBean.classEmployee.classDepartment.class
C. Emp.earemp-ejb.jarMETA-INF/persistence.cmlEmployee.classDepartment.classEmployeeService.classEmployeeServiceBean.clas s
D. Emp.earemp-ejb.jarpersistence.xmlEmployee.classDepartment.classEmployeeService.classEmployeeServiceBean.cl ass

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:

QUESTION 26
Assume you have been tasked with building an ejb-jar containing an EJB application. The EJB application contains local, remote, and web service and-point EJBs that provide reusable services within an enterprise. When the application is deployed clients will access the remote session beans using the global JNDI name java/: ServiceLayer / <bean_name>. All of the EJBs are located in the com.acme.servicelayer package and are deployed as class files. The application uses a maximum of deployment descriptor and annotation configuration?

A. Name the jar servicelayer.jar with the following structure: / (Root)I META INF /I MANIFEST.MF I classes/I com/I acme/I servicelayer /<list of classes>
B. Name the jar servicelayer.jar with the following structure: / (Root)I META INF/I MANIFEST.MFI ejb - jar.xmlI classes/I com/I acme /I servicelayer/<list of classes>

C. Name the jar servicelayer.jar with the following structure: /(Root)l META INF /l ejb jar.xmll com/l acme/l servicelayer/<list of classes>
D. Name the jar servicelayer.jar with the following structure: / (Root)l META INF/l MANIFEST.MFl ejb-jar.xmll com/l servicelayer/<list of classes>

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
References: The Java EE 6Tutorial, Packaging Enterprise Beans in EJB JAR Modules

QUESTION 27
A developer writes a stateful session bean with local business interface Bar containing method test.
Method test is implemented as:

```
11. @Remove
12. public void test () {}
```

A business method in a stateless session bean invokes a reference to bean Bar as follows:

```
11. @EJB Bar bar;
12.
13. public void foo () {
14. bar.test ();
15. bar.test();
16. }
```

Assuming execution reaches Line 15, what is the expected result?

A. Method foo returns without error.
B. A javax.ejb.NoSuchEJBException is thrown.
C. A java.rmi.NoSuchObjectException is thrown.
D. A javax.ejb.NoSuchEntityException is thrown.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

QUESTION 28
MyMsg is a JMS message-driven bean with container-managed transaction demarcation. FooBean is an EJB 3.x stateless session bean that sends message to the JMS destination with MyMsgBean is associated.

MyMsgBean's message listener method has transaction attribute REQUIRED, and is defined as follows:

```
10. public class MyMsgBean implements javax.jms.messageListener {
11. public void onMessage(javax.jms.Message message) {
12. / / do some work not shown here
13. thrown new RuntimeException("unexpected error . . . ");
14. }
```

Which statement is true about the result of message processing?

A. FooBean receives javax.ejb.EJBException.
B. The container discards the MyMsgBean bean instance.

C. FooBean receives the original RuntimeException thrown from the message listener method.
D. The container does NOT roll back the transaction, and FooBean can continue the transaction.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

QUESTION 29
Given an JMS message-driven bean, which statement is true about its exception handling?

A. Its message listener method must NOT throw any checked exception.
B. Its message listener method can throw java.rmi.RemoteException.
C. Its message listener method can throw any checked exception except java.rmi.RemoteException.
D. Its message listener method can throw any checked exception that implements java.io.Serializable.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 30
Which is a correct way to define a runtime exception as an EJB 3.x application exception?

A. public class MyAppException extends javax.ejb.EJBException
B. @ApplicationExceptionpublic class MyAppException extends javax.ejb.EJBException
C. public class MyAppException extends javax.lang.EJBException
D. @ApplicationExceptionpublic class MyAppException extends javax.lang.EJBException

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Use the @javax.ejb.ApplicationException annotation to specify that an exception class is an application exception thrown by a business method of the EJB. The EJB container reports the exception directly to the client in the event of the application error.

Note:
java.lang.Object
java.lang.Throwable
java.lang.Exception
java.lang.RuntimeException
javax.ejb.EJBException
javax.ejb
public class EJBException
extends java.lang.RuntimeException
The EJBException is thrown to report that the invoked business method or callback method could not be completed because of an unexpected error (e.g. the instance failed to open a database connection).

Example:
The following ProcessingException.java file shows how to use the @ApplicationException annotation to specify

that an exception class is an application exception thrown by one of the business methods of the EJB:
package examples;
import javax.ejb.ApplicationException;
/** * Application exception class thrown when there was a processing error * with a business method of the EJB. Annotated with the * @ApplicationException annotation. */ @ApplicationException()public class ProcessingException extends Exception {

QUESTION 31
A developer writes a stateless session bean with one local business interface and with container- managed transactions. All business methods have transaction attribute REQUIRED. The bean has an injected field sessionCtx of the type SessionContext. Which two operations are allowed in a business method of the bean? (Choose two.)

A. sessionCtx. getEJBObject
B. sessionCtx.setRollbackOnly
C. sessionCtx. getMessageContext
D. sessionCtx. getBusinessObject
E. sessionCtx. getEJBLocalObject

Correct Answer: BD
Section: (none)
Explanation

Explanation/Reference:

QUESTION 32
Which two are programming restrictions in the EJB specification? (Choose two.)

A. An enterprise bean must NOT attempt to load a native library.
B. An enterprise bean must NOT declare static fields as final.
C. An enterprise bean must NOT attempt to create a new security manager.
D. An enterprise bean must NOT propagate a RuntimeException to the container.
E. An enterprise bean must NOT attempt to obtain a javax.naming.InitialContext.

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The following is a list of Java features that you should avoid, hence restricting their use in your EJB components' implementation code:
(A) Loading native libraries.
(C) Attempting to create or obtain a class loader, set or create a new security manager (C), stop the JVM, change the input, output, and error streams. That restriction enforces security and maintains the EJB container's ability to manage the runtime environment.

(not B) Using static, nonfinal fields. Declaring all static fields in the EJB component as final is recommended. That ensures consistent runtime semantics so that EJB containers have the flexibility to distribute instances across multiple JVMs.

QUESTION 33
Which three methods can Bean Developer use in any EJB compliant container implementing the full Java EE6 product? (Choose three.)

A. Class.getClassLoader

B. FileInputStream.read
C. DataSource.getConnection
D. EJBContext.getCallerPrincipal
E. Executors.newSingleThreadExecutor
F. QueueConnection.createQueueSession

Correct Answer: CDF
Section: (none)
Explanation

Explanation/Reference:

QUESTION 34
Which two annotations can be applied at the class, method, and field levels? (Choose two.)

A. @EJB
B. @Init
C. @Resource
D. @RolesAllowed
E. @PostActivate

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation:
A: javax.ejb.EJB
Description

Target: Class, Method, Field

Specifies a dependency or reference to an EJB business or home interface.

You annotate a bean's instance variable with the @EJB annotation to specify a dependence on another EJB.

C: javax.annotation.Resource
Description

Target: Class, Method, Field

Specifies a dependence on an external resource, such as a JDBC data source or a JMS destination or connection factory.

Incorrect:
B: javax.ejb.Init
Description
Target: Method

D: javax.annotation.security.RolesAllowed
Description
Target: Class, Method
Specifies the list of security roles that are allowed to access methods in the EJB.

E: javax.ejb.PostActivate
Description

Target: Method
Specifies the lifecycle callback method that signals that the EJB container has just reactivated the bean instance.

QUESTION 35
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.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 36
Which must result in the destruction of a stateful session bean?

A. A client calls an @Remove method and the method returns successfully.
B. The server in which the stateful session bean was created is restarted.
C. The stateful session bean participates in a transaction that is rolled back.
D. The stateful session bean is chosen as a last recently used (LRU) victim for passivation.

Correct Answer: A
Section: (none)
Explanation

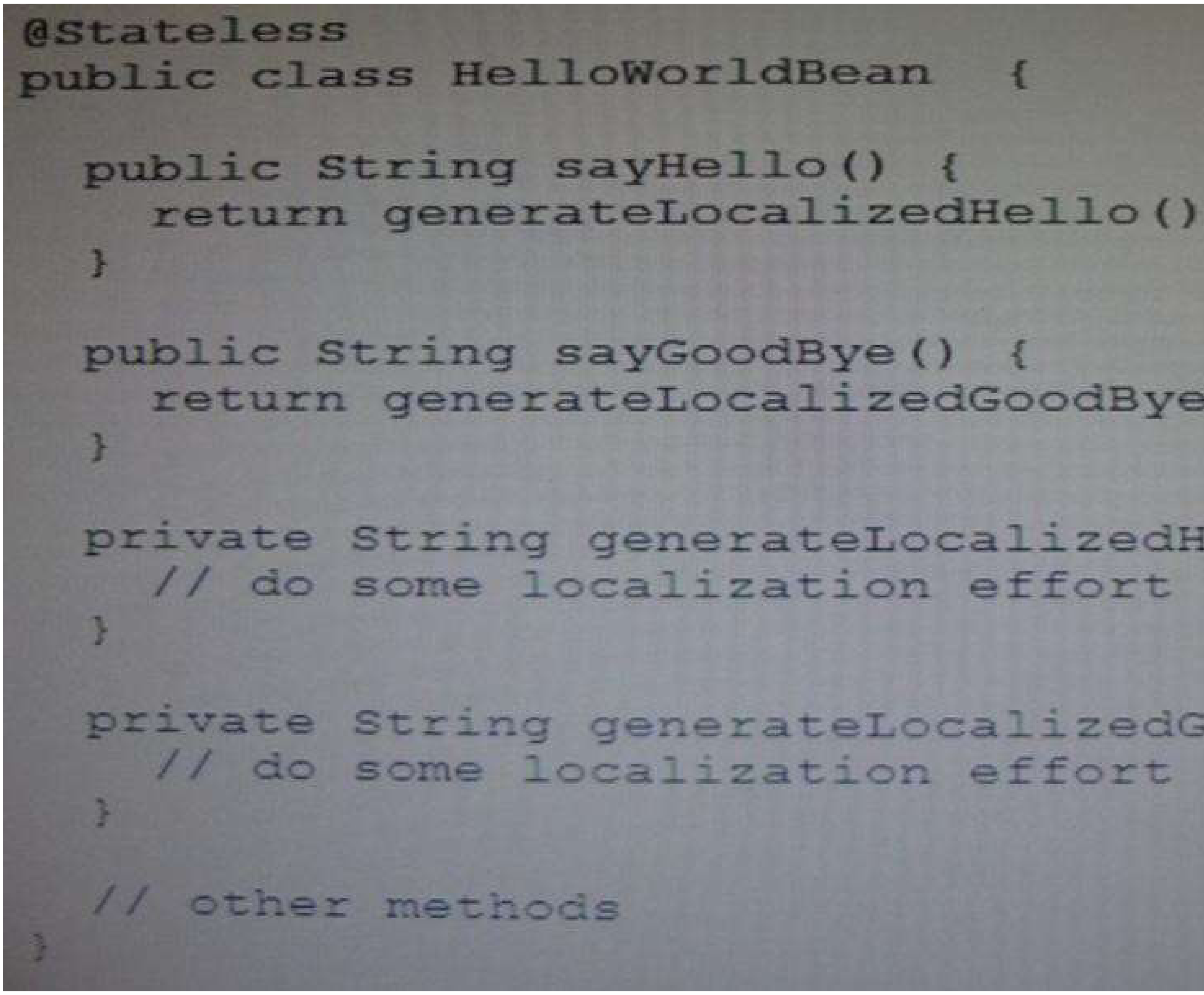
Explanation/Reference:
Explanation:
While in the ready stage, the EJB container may decide to deactivate, or passivate, the bean by moving it from memory to secondary storage. (Typically, the EJB container uses a least-recently-used algorithm to select a bean for passivation.) The EJB container invokes the method annotated @PrePassivate, if any, immediately before passivating it. If a client invokes a business method on the bean while it is in the passive stage, the EJB container activates the bean, calls the method annotated @PostActivate, if any, and then moves it to the ready stage.

Note:

At the end of the lifecycle, the client invokes a method annotated @Remove, and the EJB container calls the method annotated @PreDestroy, if any. The bean's instance is then ready for garbage collection.
* When a stateful bean is passivated, the instance fields are read and then written to the secondary storage associated with the EJB object. When the stateful session bean has been successfully passivated, the instance is evicted from memory; it is destroyed.

* When a passivated bean instance times out or when a client invokes the method marked with @Remove, the container may destroy the bean. Before destroying, the container will invoke the method annotated with @PreDestroy.

QUESTION 37
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 generateLocalizedH
    // do some localization effort
  }

  private String generateLocalizedG
    // do some localization effort
  }

  // other methods
}
```

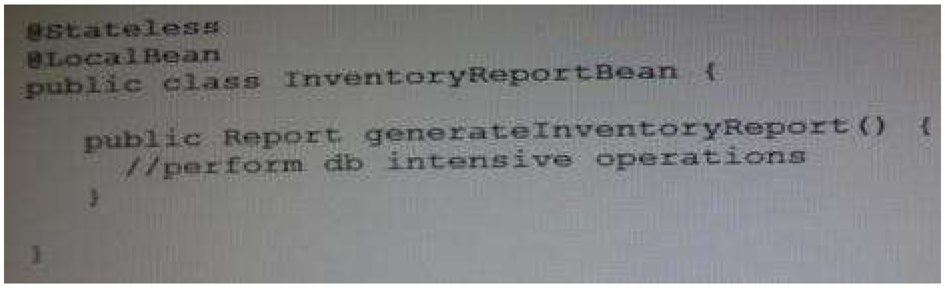
Which types of clients are guaranteed to have access to HelloWorldBean:

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

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

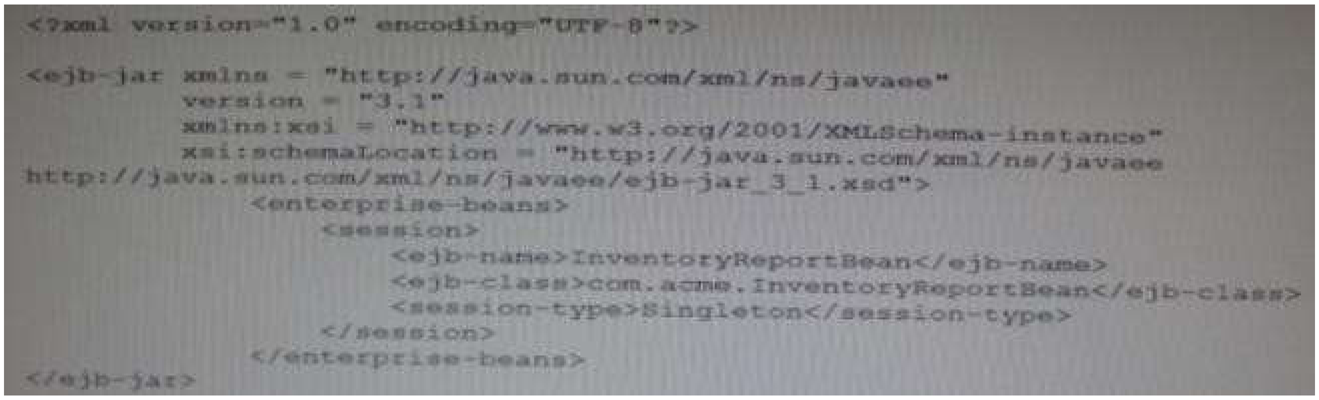
QUESTION 38
Assume an EJB application is comprised of the following EJB fragment:



```
@Stateless
@LocalBean
public class InventoryReportBean {

    public Report generateInventoryReport() {
      //perform db intensive operations
    }

}
```

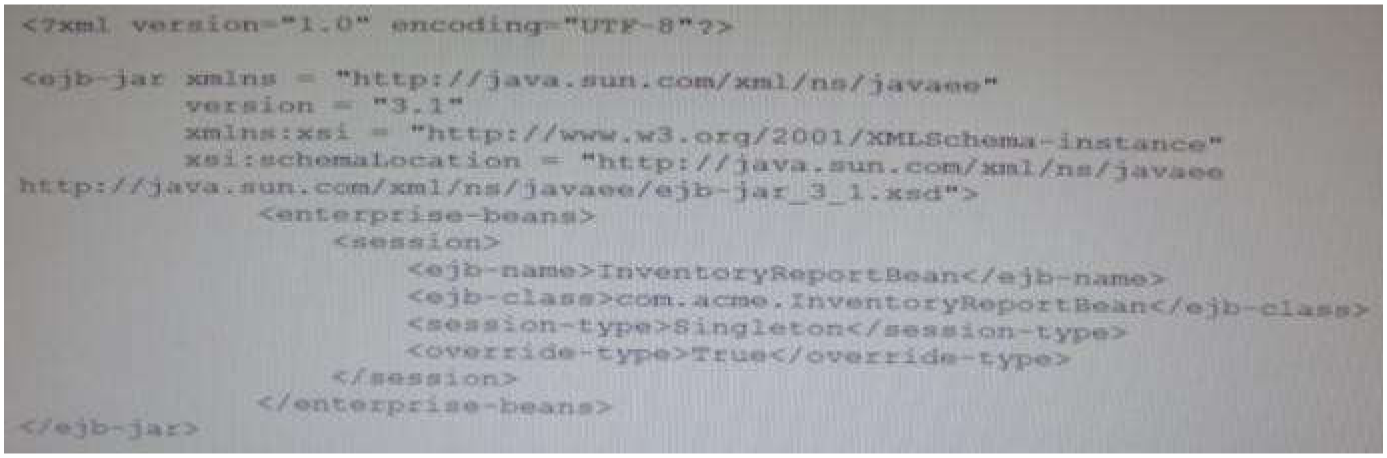
You have been asked to convert the type of InventoryReportBean into a singleton session bean. How would you achieve this task?

Exhibit C:



```
<?xml version="1.0" encoding="UTF-8"?>

<ejb-jar xmlns = "http://java.sun.com/xml/ns/javaee"
         version = "3.1"
         xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation = "http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/ejb-jar_3_1.xsd">
            <enterprise-beans>
                <session>
                    <ejb-name>InventoryReportBean</ejb-name>
                    <ejb-class>com.acme.InventoryReportBean</ejb-class>
                    <session-type>Singleton</session-type>
                </session>
            </enterprise-beans>
</ejb-jar>
```

Exhibit D:



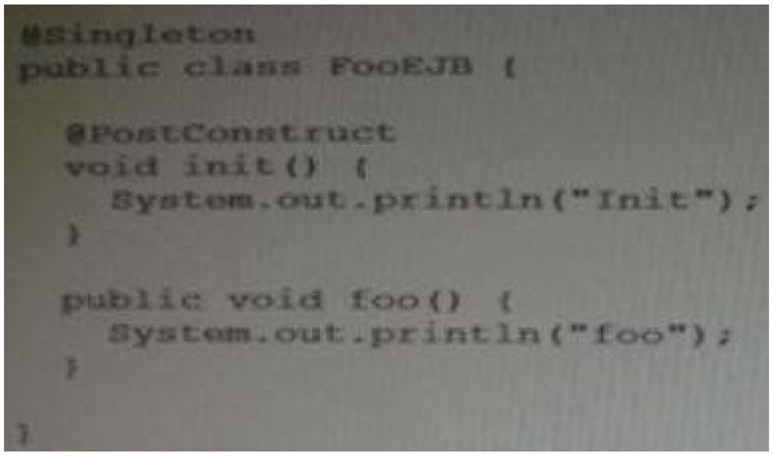
```
<?xml version="1.0" encoding="UTF-8"?>

<ejb-jar xmlns = "http://java.sun.com/xml/ns/javaee"
         version = "3.1"
         xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation = "http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/ejb-jar_3_1.xsd">
            <enterprise-beans>
                <session>
                    <ejb-name>InventoryReportBean</ejb-name>
                    <ejb-class>com.acme.InventoryReportBean</ejb-class>
                    <session-type>Singleton</session-type>
                    <override-type>True</override-type>
                </session>
            </enterprise-beans>
</ejb-jar>
```

A. Keep InventoryReportBean as it is, modifying the internal structure to function as a singleton
B. Change the @Stateless annotation of InventoryReportBean to @Singleton
C. Create an ejb-jar.xml file, and override the annotation configuration information as in exhibit C above.
D. Create an ejb-jar.xml file, and override the annotation configuration information as in exhibit D above.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

QUESTION 39
Given Singleton bean FooEJB:


```
@Singleton
public class FooEJB {

   @PostConstruct
   void init() {
      System.out.println("Init");
   }

   public void foo() {
      System.out.println("foo");
   }

}
```

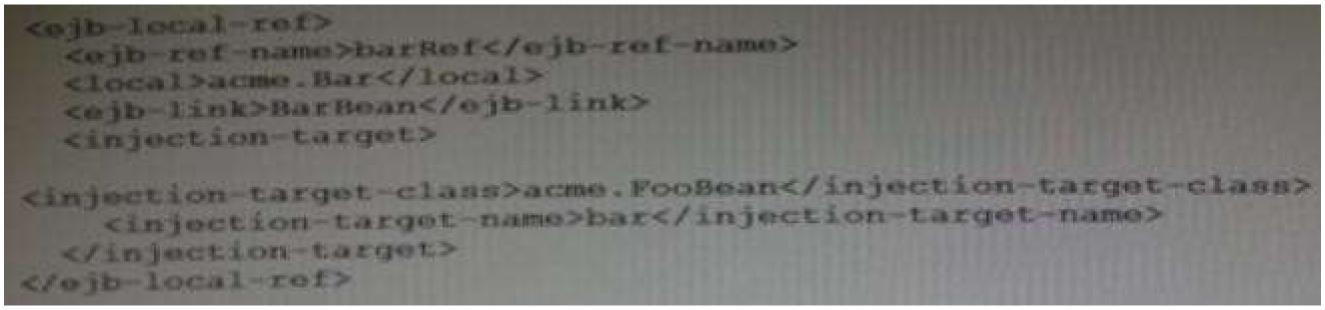
FooEJB is packaged as the only bean in an ejb-jar and deployed to a server instance. Which represents the output generated from FooEJB after the deployment has completed?

A. Init
B. foo
C. Init foo
D. <no output>
E. a or d

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

QUESTION 40
A developer writes a stateless session bean FooBean and uses its deployment descriptor to declare a local ejb dependency on a stateful session bean in the same ejb-jar.


```
<ejb-local-ref>
  <ejb-ref-name>barRef</ejb-ref-name>
  <local>acme.Bar</local>
  <ejb-link>BarBean</ejb-link>
  <injection-target>
<injection-target-class>acme.FooBean</injection-target-class>
    <injection-target-name>bar</injection-target-name>
  </injection-target>
</ejb-local-ref>
```

Which environment annotation, when declared within the FooBean bean class, is equivalent to the ejb- local-ref shown above?

A. @EJB(beanName="BarBean")Private acme.Bar barRef;
B. @EJB(name="bar", beanName="BarBean")Private acme.Bar barRef;

C. @EJB(name="barRef", beanName="BarBean")Private acme.Bar bar;
D. @EJB(name="ejab/barRef", beanName="BarBean")Private acme.Bar bar;

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
name is barRef

Example:

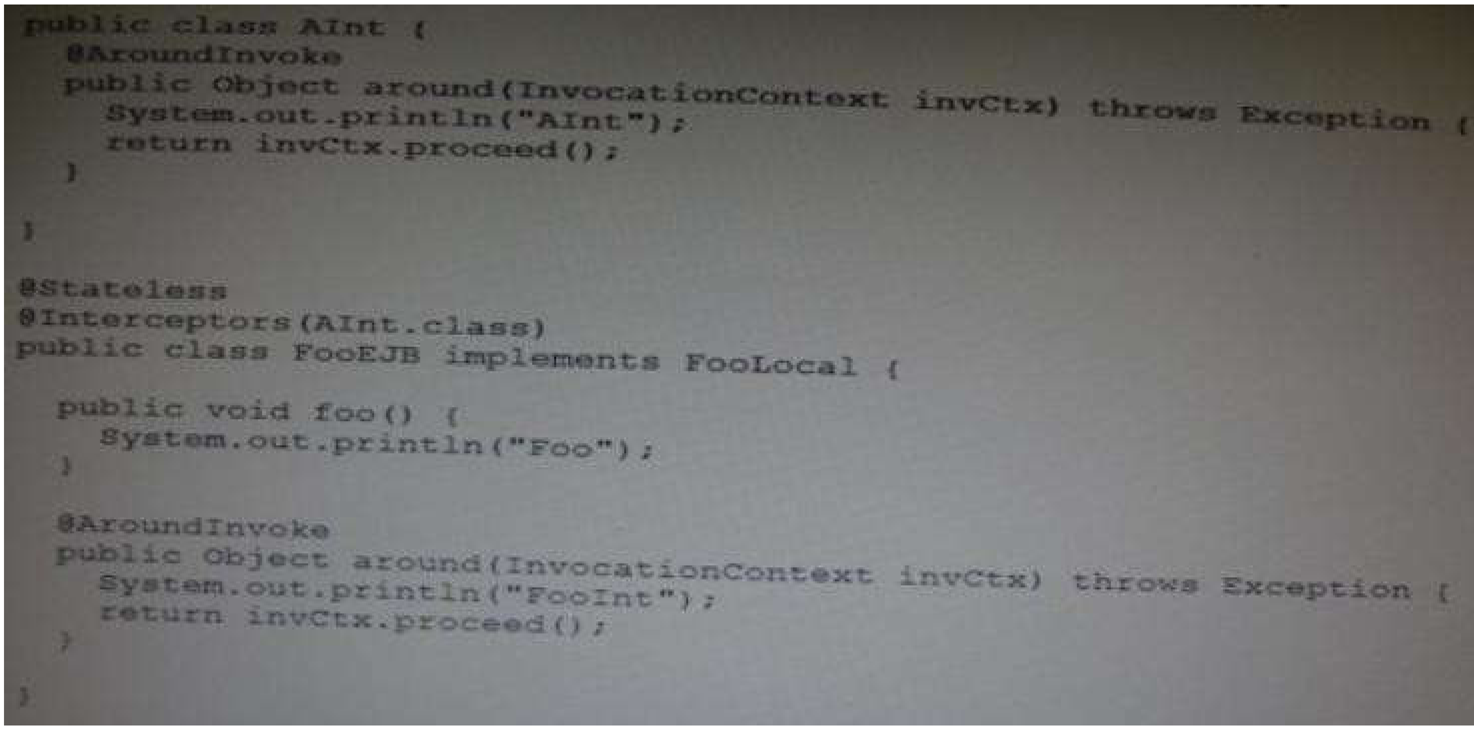
ejb-local-ref
share [gp] share [fb] share [tw] contribute
Via annotation
Usable by EJB, Interceptor, Servlet, Filter, or Listener package org.superbiz.refs;

```
import javax.ejb.EJB;
import javax.ejb.Stateless;
import javax.naming.InitialContext;

@Stateless
@EJB(name = "myFooEjb", beanInterface = FooLocal.class) public class MyEjbLocalRefBean implements
MyBeanInterface {

@EJB
private BarLocal myBarEjb;

public void someBusinessMethod() throws Exception {
if (myBarEjb == null) throw new NullPointerException("myBarEjb not injected");

// Both can be looked up from JNDI as well
InitialContext context = new InitialContext();
FooLocal fooLocal = (FooLocal) context.lookup("java:comp/env/myFooEjb"); BarLocal barLocal = (BarLocal)
context.lookup("java:comp/env/org.superbiz.refs.MyEjbLocalRefBean/myBarEjb"); }
}
```

Via xml
The above @EJB annotation usage is 100% equivalent to the following xml.

```
<ejb-local-ref>
<ejb-ref-name>myFooEjb</ejb-ref-name>
<local>org.superbiz.refs.FooLocal</local>
</ejb-local-ref>

<ejb-local-ref>
<ejb-ref-name>org.superbiz.refs.MyEjbLocalRefBean/myBarEjb</ejb-ref-name>
<local>org.superbiz.refs.BarLocal</local>
<injection-target>
<injection-target-class>org.superbiz.refs.MyEjbLocalRefBean</injection-target-class> <injection-target-
name>myBarEjb</injection-target-name> </injection-target>
</ejb-local-ref>
```

QUESTION 41
A developer writes an interceptor class and a stateless session bean:



```
public class AInt {
  @AroundInvoke
  public Object around(InvocationContext invCtx) throws Exception {
    System.out.println("AInt");
    return invCtx.proceed();
  }

}

@Stateless
@Interceptors(AInt.class)
public class FooEJB implements FooLocal {

  public void foo() {
    System.out.println("Foo");
  }

  @AroundInvoke
  public Object around(InvocationContext invCtx) throws Exception {
    System.out.println("FooInt");
    return invCtx.proceed();
  }

}
```

A client acquires an EJB reference to the FooLocal business interface and invokes the foo() method one time. Which describes the output?

A. Foo FooInt AInt
B. AInt Foo
C. AInt FooInt Foo
D. FooInt AInt Foo

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
* At the end of the chain of interceptors, the actual bean method gets called.

* Interceptors can be bound in three different ways:

Default
Class level
Method level

In this question both class level and method level interceptors are used. The class level interceptor intercepts before the method-level interceptor.

Note:

* Interceptors are used in conjunction with Java EE managed classes to allow developers to invoke interceptor methods on an associated target class, in conjunction with method invocations or lifecycle events. Common uses of interceptors are logging, auditing, and profiling.

* An interceptor can be defined within a target class as an interceptor method, or in an associated class called an interceptor class. Interceptor classes contain methods that are invoked in conjunction with the methods or

lifecycle events of the target class.

Interceptor classes and methods are defined using metadata annotations, or in the deployment descriptor of the application containing the interceptors and target classes.

* javax.interceptor.AroundInvoke
Designates the method as an interceptor method.

* The target class can have any number of interceptor classes associated with it. The order in which the interceptor classes are invoked is determined by the order in which the interceptor classes are defined in the javax.interceptor.Interceptors annotation.

QUESTION 42
A developer writes an interceptor class containing an AroundInvoke method, and applies it to the local business interface method of a stateless session bean:

```
11. @Interceptors(FooInterceptor.class)
12. public void bar() ()
```

A client obtains a reference to the bean's local business interface, and calls the method bar two times from the same thread. Assuming that the container dispatches both cell to the same stateless session bean instance, how many instances of the FooInterceptor class will be used?

A. 0
B. 2
C. 1
D. Either 1 or 2

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
You can specify one nonbusiness method as the interceptor method for a stateless or stateful session bean. Each time a client invokes a session bean business method, OC4J intercepts the invocation and invokes the interceptor method.

QUESTION 43
While excepting a business method in a stateless session bean the container rolls back the method's transaction. Which three are possible causes for the container's behavior? (Choose three.)

A. The bean uses container-managed transactions and invokes EJBContext.setRollbackOnly.
B. The bean uses container-managed transactions and invokes EJBContext.getRollbackOnly.
C. The business method throws a java.lang.NullPointerException.
D. The business method throws a checked exception of a class type that is marked with the @ApplicationException annotation with the rollback element value true.
E. The business method throws a unchecked exception of a class type that is marked with the @ApplicationException annotation with the rollback element value true.
F. The bean uses container-managed transactions and throws a checked exception of a class type that is marked with the @ApplicationException annotation with the rollback element value false.

Correct Answer: ADF
Section: (none)
Explanation

Explanation/Reference:

Explanation:
A: setRollbackOnly

Mark the current transaction for rollback. The transaction will become permanently marked for rollback. A transaction marked for rollback can never commit. Only enterprise beans with container- managed transactions are allowed to use this method.

Note:
* In a stateless session bean with bean-managed transactions, a business method must commit or roll back a transaction before returning.

* Bean-Managed Transactions
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.
* (incorrect) Unchecked runtime exceptions represent conditions that, generally speaking, reflect errors in your program's logic and cannot be reasonably recovered from at run time.

Incorrect:
B: getRollbackOnly

Test if the transaction has been marked for rollback only. An enterprise bean instance can use this operation, for example, to test after an exception has been caught, whether it is fruitless to continue computation on behalf of the current transaction. Only enterprise beans with container-managed transactions are allowed to use this method.

QUESTION 44
A stateless session bean FooBean implements an asynchronous business method foo() on its bean class:

@Asynchronous
public void foo() (...)

The asynchronous business method is exposed through a remote business interface FooRemote. A caller acquires an EJB reference to this bean and invokes it as follows:

100. fooRemoteRef.foo();

Which exception can result from the invocation on line 100?

A. java.rmi.RemoteException
B. java.util.concurrent.ExecutionException
C. javax.ejb.EJBException
D. java.lang.IllegalArgumentException

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

Note:
* RemoteRef represents the handle for a remote object. A RemoteStub uses a remote reference to carry out a remote method invocation to a remote object.
* invoke

public Object invoke(Remote obj,

Method method,
Object[] params,
long opnum)
throws Exception
Invoke a method. This form of delegating method invocation to the reference allows the reference to take care of setting up the connection to the remote host, marshaling some representation for the method and parameters, then communicating the method invocation to the remote host. This method either returns the result of a method invocation on the remote object which resides on the remote host or throws a RemoteException if the call failed or an application-level exception if the remote invocation throws an exception.
Parameters:

QUESTION 45
A developer impalements an asynchronous implementation for calculating insurance proposals. The input data for the calculations is made available on a single message queue. Two types of insurance proposals will be calculated: car and life. Message with data for other insurance types are posted on the queue but should be left on the queue by this implementation.
Which statement is true?

A. The developer will NOT succeed because all messages will be consumed from the queue.
B. The developer can implement a push-back mechanism if the message is of the wrong type.
C. The developer can use a messageSelector to receive only the car and life data message if the JMS body contains selectable data.
D. The developer can use a messageSelector to receive only the car and life data message if the header contains properties to make selection.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
A JMS message selector allows a client to specify, by header field references and property references, the messages it is interested in. Only messages whose header and property values match the selector are delivered. What it means for a message not to be delivered depends on the MessageConsumer being used (see QueueReceiver and TopicSubscriber).

QUESTION 46
Which two statements are true JMS message-driven beans? (Choose two.)

A. The developer can use JMS message selector declarations to restrict the message that the bean receives.
B. The developer can associate the bean with a specific queue or topic using the resource-ref element of the deployment descriptor.
C. To achieve concurrent processing of more than one message at a time, more than one bean class must be associated with the same JMS queue.
D. The developer can use the activationConfig element of the MessageDriven annotation to specify whether the bean should be associated with a queue or a topic.

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation:
A: Elements in the deployment descriptor

The description of a MDB (message-driven beans) in the EJB 2.0 deployment descriptor contains the following specific elements:

* the JMS acknowledgement mode: auto-acknowledge or dups-ok-acknowledge

* an eventual JMS message selector: this is a JMS concept which allows the filtering of the messages sent to the destination

* a message-driven-destination, which contains the destination type (Queue or Topic) and the subscription

D: Example:
The following example is a basic message-driven bean:

```
@MessageDriven(activationConfig={
@ActivationConfigProperty(propertyName="destination", propertyValue="myDestination"),
@ActivationConfigProperty(propertyName="destinationType", propertyValue="javax.jms.Queue") })
public class MsgBean implements javax.jms.MessageListener {

public void onMessage(javax.jms.Message msg) {

String receivedMsg = ((TextMessage) msg).getText();
System.out.println("Received message: " + receivedMsg);

}

}
```

QUESTION 47
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.

Correct Answer: BD
Section: (none)
Explanation

Explanation/Reference:
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.

QUESTION 48
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)

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
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 49
You are writing a client that sends a message to a JMS queue.
What two statements are true?

A. You cannot use resource injection to access a JMS destination from a Java EE application client.
B. You can use resource injection to access a JMS destination from a servlet.
C. You must use a JNDI lookup to access a JMS destination from a standalone Java class.
D. You cannot use a JNDI lookup to access a JMS destination from a session bean.

Correct Answer: BC
Section: (none)
Explanation

Explanation/Reference:
Explanation:
B: In addition to injecting a connection factory resource into a client program, you usually inject a destination resource. Unlike connection factories, destinations are specific to one domain or the other.

Note:
* A destination is the object a client uses to specify the target of messages it produces and the source of messages it consumes. In the PTP messaging domain, destinations are called queues. In the pub/sub messaging domain, destinations are called topics.

* In addition to looking up a connection factory in a client program, you usually look up a destination. Unlike connection factories, destinations are specific to one domain or the other. To create an application that allows you to use the same code for both topics and queues, you cast and assign the destination to a Destination object. To preserve the semantics of queues and topics, however, you cast and assign the object to a destination of the appropriate type.

For example, the following line of code performs a JNDI lookup of the previously created topic jms/MyTopic and casts and assigns it to a Destination object:

Destination myDest = (Destination) ctx.lookup("jms/MyTopic"); The following line of code looks up a queue named jms/MyQueue and casts and assigns it to a Queue object:
Queue myQueue = (Queue) ctx.lookup("jms/MyQueue");

QUESTION 50
Assume you have been tasked with building an ejb-jar containing an EJB application. The EJB application

contains local, remote, and web service end-point EJBs that provide reusable sevices within an enterprise. When the application is deployed, client will access the remote session beans using the global JNDI name java:ServiceLayer/<bean_name>. All of the EJBs are located in the com.acme.sevicelayer package and are deployed as class files. The application uses a mixture of deployment descriptor and annotation configuration.

Which JAR representation best represents deployed structure for the previous requirements?

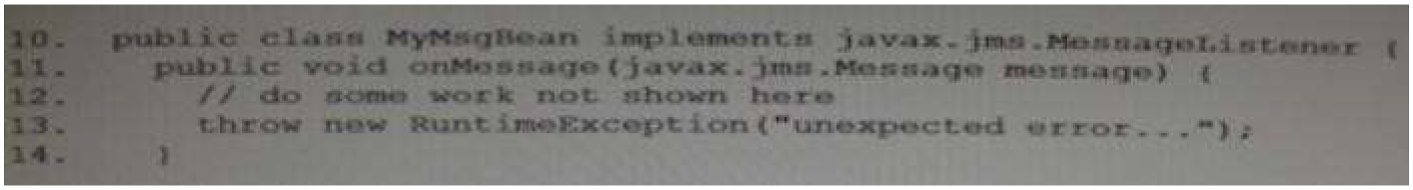
A. Option A
B. Option B
C. Option C
D. Option D

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

QUESTION 51
MyMsgBean is a JMS message-driven with container-managed transaction demarcation. FooBean is an EJB 3.x stateless session bean that sends messages to the JMS destination with which MyMsgBean is associated.

MyMsgBean's message listener method has transaction attribute REQUIRED, and is defined as follows:



```
public class MyMsgBean implements javax.jms.MessageListener {
  public void onMessage(javax.jms.Message message) {
    // do some work not shown here
    throw new RuntimeException("unexpected error...");
  }
```

Which statement is true about the result of the message processing?

A. FooBean receives javax.ejb.EJBException.
B. The container discards the MyMsgBean bean instance.
C. FooBean receives the original RuntimeException thrown from the message listener method.
D. The container does NOT roll back the transaction, and FooBean can continue the transaction.

Correct Answer: B
Section: (none)
Explanation

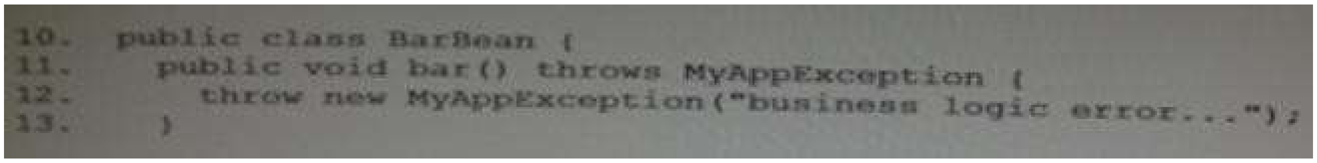
Explanation/Reference:
Explanation:

Note:
* Required Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the method executes within the client's transaction. If the client is not associated with a transaction, the container starts a new transaction before running the method.

The Required attribute is the implicit transaction attribute for all enterprise bean methods running with container-managed transaction demarcation. You typically do not set the Required attribute unless you need to override another transaction attribute. Because transaction attributes are declarative, you can easily change them later.

QUESTION 52
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.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
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.

QUESTION 53
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

Correct Answer: ADF
Section: (none)
Explanation

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

QUESTION 54

A developer examines a list of potential enterprise applications and selects the most appropriate technologies to use for each application.

For which two applications is EJB an appropriate solution? (Choose two.)

A. To render a GUI for mobile clients.
B. As a container for web-tier components including JSP.
C. As a Web service endpoint accessed by non-Java clients.
D. To receive and respond to HTTP Post requests directly from a web browser.
E. As an online shopping cart which can persist across multiple sessions with a single client.

Correct Answer: CE
Section: (none)
Explanation

Explanation/Reference:

QUESTION 55
Which two statements are true? (Choose two.)

A. Typically, remotely accessible objects should be coarse-grained.
B. If a client accesses an enterprise bean locally such access must be mediated by the EJB container.
C. A given enterprise bean's transaction information is immutable because it is deployed across various containers.
D. If a container provides services NOT required by the EJB specification, then that container is NOT considered to be an EJB container.
E. An enterprise bean's transaction Information can be accessed by external tools only if the information is contained in an XML deployment descriptor.

Correct Answer: AB
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Enterprise JavaBeans 3.1, Final Release: 3.2.3 Choosing Between a Local or Remote Client View Remote calls are potentially expensive. They involve network latency, overhead of the client and server software stacks, argument copying, etc. Remote calls are typically programmed in a coarse-grained manner with few interactions between the client and bean.21.3 Container Provider's ResponsibilityThis section defines the container's responsibilities for providing the runtime environment to the enterprise bean instances. The requirements described here are considered to be the minimal requirements; a container may choose to provide additional functionality that is not required by the EJB specification.

QUESTION 56
Assume you would like to receive notification from the container as a stateless session bean transitions to and from the ready state.

Which of the following life cycle back annotations would you use? (Choose one.)

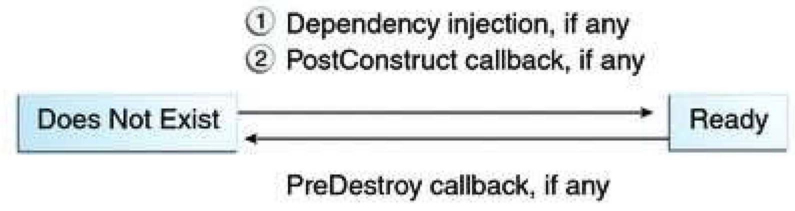
A. @PostConstruct, @PostDestroy
B. @PostConstruct, @PreDestroy
C. @PreConstruct, @PostDestroy
D. @PostConstruct, @PostDestroy, @Remove
E. @PostConstruct, @PreDestroy, @Remove

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The Lifecycle of a Stateless Session Bean
The EJB container typically creates and maintains a pool of stateless session beans, beginning the stateless session bean's lifecycle. The container performs any dependency injection and then invokes the method annotated @PostConstruct, if it exists. The bean is now ready to have its business methods invoked by a client.

At the end of the lifecycle, the EJB container calls the method annotated @PreDestroy, if it exists. The bean's instance is then ready for garbage collection.

Lifecycle of a Stateless Session Bean:


① Dependency injection, if any
② PostConstruct callback, if any
Does Not Exist
Ready
PreDestroy callback, if any

Note: An enterprise bean goes through various stages during its lifetime, or lifecycle. Each type of enterprise bean (stateful session, stateless session, singleton session, or message-driven) has a different lifecycle.

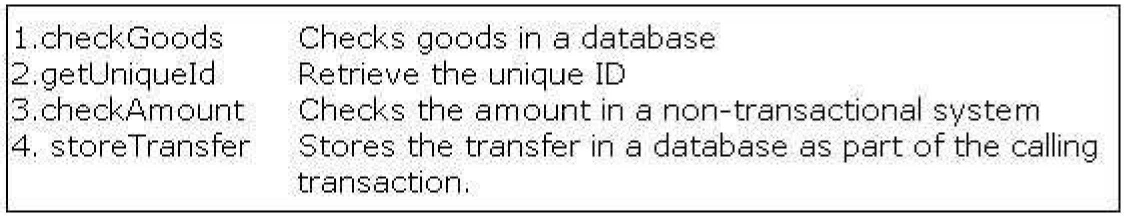
QUESTION 57
Which API must an EJB 3.1 container make available to enterprise beans at runtime?

A. The JXTA 1.1 API
B. The MIDP 2.0 API
C. The Java SE 6 JNDI API
D. The Java SE 5 JDBC API

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 58
A developer implements a system in which transfers of goods are monitored. Each transfer needs a unique ID for tracking purposes. The unique ID is generated by an existing system which is also used by other applications. For performance reasons, the transaction that gets the unique ID should be as short as possible. The scenario is implemented in four steps which are implemented in four business methods in a CMT session bean:



1.checkGoods	Checks goods in a database
2.getUniqueId	Retrieve the unique ID
3.checkAmount	Checks the amount in a non-transactional system
4. storeTransfer	Stores the transfer in a database as part of the calling transaction.

These methods are called by the addTransfer method of a second CMT session bean in the following order:

checkGooods, getUniqueId, checkAmount, storeTranfer

Assuming no other transaction-related metadata, which is the correct set of transaction attributes for the methods in the session beans?

A. 0.addTransferREQUIRED1.checkGoodsREQUIRED2.getUnigueIdREQUIRES_NEW3.checkAmounts NOT_SUPPORTED4.storeTransferMANDATORY
B. 0.addTransferREQUIRED1.checkGoodsREQUIRED2.getUnigueIdREQUIRED3.checkAmountsREQ UIRED4.storeTransferREQUIRED
C. 0.addTransferREQUIRED1.checkGoodsREQUIRED2.getUnigueIdREQUIRES_NEW3.checkAmounts NEVER4.storeTransferMANDATORY
D. 0.addTransferNOT_SUPPORTED1.checkGoodsREQUIRED2.getUnigueIdREQUIRES_NEW3.check AmountsNOT_SUPPORTED4.storeTransferMANDATORY

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

Step 2: Must start a new transaction. use REQUIRES_NEW

Step 3: No need for this step: use Not Supported
Use the NotSupported attribute for methods that don't need transactions. Because transactions involve overhead, this attribute may improve performance.

Step 4: Use Mandatory:
Use the Mandatory attribute if the enterprise bean's method must use the transaction of the client.

Note:
* In an enterprise bean with container-managed transaction (CMT) demarcation, the EJB container sets the boundaries of the transactions. You can use container-managed transactions with any type of enterprise bean: session, or message-driven. Container-managed transactions simplify development because the enterprise bean code does not explicitly mark the transaction's boundaries. The code does not include statements that begin and end the transaction.

* A transaction attribute can have one of the following values:

Required

RequiresNew

Mandatory

NotSupported

Supports

Never

* Required Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the method executes within the client's transaction. If the client is not associated with a transaction, the container starts a new transaction before running the method.

The Required attribute is the implicit transaction attribute for all enterprise bean methods running with container-managed transaction demarcation. You typically do not set the Required attribute unless you need to override another transaction attribute. Because transaction attributes are declarative, you can easily change them later.

* RequiresNew Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the container takes the following steps:

Suspends the client's transaction

Starts a new transaction

Delegates the call to the method

Resumes the client's transaction after the method completes If the client is not associated with a transaction, the container starts a new transaction before running the method.

You should use the RequiresNew attribute when you want to ensure that the method always runs within a new transaction.

* Mandatory Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the method executes within the client's transaction. If the client is not associated with a transaction, the container throws the TransactionRequiredException.

Use the Mandatory attribute if the enterprise bean's method must use the transaction of the client.

* NotSupported Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the container suspends the client's transaction before invoking the method. After the method has completed, the container resumes the client's transaction.

If the client is not associated with a transaction, the container does not start a new transaction before running the method.

Use the NotSupported attribute for methods that don't need transactions. Because transactions involve overhead, this attribute may improve performance.

QUESTION 59
A developer implements a CMT session bean with a method storeBoth which inserts data both a related database and an LDAP server. The relational database supports transactions while the LDAP system does NOT.

Given that both updates should succeed or be rolled back, while is the best solution?

A. Implement the SessionSynchoronization interface in the session bean. In the afterCompleteion method, the LDAP inserts are rolled back if false is passed as an argument to the afterCompletion method.
B. Define the transaction attribute of the method storeBoth as REQUIRED. The container manages the transactions and will roll back modifications if something goes wrong in either database insert or LDAP insert.
C. Define the transaction attribute of the method storeBoth as REQUIRED_NEW. Carry out the database insert first. Subsequently, execute the LDAP inserts, catching LDAP exceptions. If exceptions are raised, call the SessionContext.setRollBackOnly method.
D. Define the transaction attribute of the method storeBoth as REQUIRED_NEW. Carry out the LDAP insert first. If SessionContext.getRollBackOnly returns false, execute the database inserts, catching SQL exceptions. If exceptions are raised, call the SessionContext.setRollBackOnly.

Correct Answer: C
Section: (none)

Explanation

Explanation/Reference:
Explanation:
The method should start a new transaction, so we use the REQUIRED_NEW attribute.

For the LDAP operation we can only detect LDAP exceptions. We cannot check the status of the LDAP operation through SessionContext.getRollBackOnly.

Note:
* CMT - Container-Managed Transactions
* RequiresNew Attribute
If the client is running within a transaction and invokes the enterprise bean's method, the container takes the following steps:

Suspends the client's transaction

Starts a new transaction

Delegates the call to the method

Resumes the client's transaction after the method completes

If the client is not associated with a transaction, the container starts a new transaction before running the method.

You should use the RequiresNew attribute when you want to ensure that the method always runs within a new transaction.

QUESTION 60
An enterprise developer has received ejb-jars from multiple Bean Provides and wants to combine them into a single ejb-jar as well as altering the method permissions on some of the beans without recompiling any of the code contained in the ejb-jar. Which is correct?

A. Bean Provide is the only role that can perform this task.
B. Deployed is the most appropriate role to perform this task.
C. Either a Deployer or System Administrator role many perform this task.
D. This problem cannot be solved using an EJB 3.x-compliant approach.
E. Application Assembler is the most appropriate role to perform this task.

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Application Assembler

The Application Assembler combines enterprise beans into larger deployable application units. The input to the Application Assembler is one or more ejb-jar files produced by the Bean Provider(s). The Application Assembler outputs one or more ejb-jar files that contain the enterprise beans along with their application assembly instructions.

Note:
* EJB Structure

The EJB Java ARchive (JAR) file is the standard format for assembling enterprise beans. This file contains the bean classes (home, remote, local, and implementation), all the utility classes, and the deployment descriptors (ejb-jar.xml and sun-ejb-jar.xml).

* The Application Assembler can also combine enterprise beans with other types of application components when composing an application.

QUESTION 61
A Java EE application server has four different security realms for user management. One of the security realms is custom made. This realm supports only individual user entries, no grouping of users, and is used by the application. Which two statements are true? (Choose two.)

A. EJB developers cannot use the isCallerInRole method.
B. The annotation @RunAs("AAA") can still be used for this application.
C. All security roles need a role-link entry in the deployment descriptor.
D. All security roles can be mapped successfully to individual users in the realm.

Correct Answer: BD
Section: (none)
Explanation

Explanation/Reference:
Explanation:

Not A, not C:
A security role reference defines a mapping between the name of a role that is called from a web component using isUserInRole(String role) and the name of a security role that has been defined for the application. If no security-role-ref element is declared in a deployment descriptor and the isUserInRole method is called, the container defaults to checking the provided role name against the list of all security roles defined for the web application. Using the default method instead of using the security-role-ref element limits your flexibility to change role names in an application without also recompiling the servlet making the call.

For example, to map the security role reference cust to the security role with role name bankCustomer, the syntax would be:

```
<servlet>

<security-role-ref>
<role-name>cust</role-name>
<role-link>bankCustomer</role-link>
</security-role-ref>

</servlet>
```

Note:
* A realm is a security policy domain defined for a web or application server. A realm contains a collection of users, who may or may not be assigned to a group.

* The protected resources on a server can be partitioned into a set of protection spaces, each with its own authentication scheme and/or authorization database containing a collection of users and groups. A realm is a complete database of users and groups identified as valid users of one or more applications and controlled by the same authentication policy.

* In some applications, authorized users are assigned to roles. In this situation, the role assigned to the user in the application must be mapped to a principal or group defined on the application server.

* A role is an abstract name for the permission to access a particular set of resources in an application. A role can be compared to a key that can open a lock. Many people might have a copy of the key. The lock doesn't care who you are, only that you have the right key.

QUESTION 62
Which is a valid use of the EJB 3.x TimerHandle object?

A. To retrieve all active timers associated with this bean
B. To adapt EJB 3.x timers to EJB 2.1 and earlier timers
C. To obtain a seralizable timer handle that may be persisted
D. To be implemented by EJB classes that are to be registered with the timer service

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The TimerHandle interface allows the bean provider to obtain a serializable timer handle that may be persisted.

Since timers are local objects, a timer handle must not be passed through a bean's remote business interface, remote interface or web service interface.

QUESTION 63
Which two statements are correct about stateless session beans? (Choose two.)

A. The bean class may declare instance variables.
B. The lifetime of the bean instance is controlled by the client.
C. The container may use the same bean instance to handle multiple business method invocations at the same time.
D. The container may use the same bean instance to handle business method invocations requested by different clients, but not concurrently.

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation:
* A: Stateless session beans are EJB's version of the traditional transaction processing applications, which are executed using a procedure call. The procedure executes from beginning to end and then returns the result. Once the procedure is done, nothing about the data that was manipulated or the details of the request are remembered. There is no state.

These restrictions don't mean that a stateless session bean can't have instance variables and therefore some kind of internal state. There's nothing that prevents you from keeping a variable that tracks the number of times a bean has been called or that tracks data for debugging. An instance variable can even hold a reference to a live resource like a URL connection for writing debugging data, verifying credit cards, or anything else that might be useful.

QUESTION 64
A developer wants to release resources within a stateless session bean class. The cleanup method should be executed by the container before an instance of the class is removed. The deployment descriptor is NOT used.

Which three statements are correct? (Choose three.)

A. The cleanup method may declare checked exceptions.
B. The cleanup method must have no arguments and return void.
C. The cleanup method is executed in an unspecified transaction and security context.
D. The developer should mark the cleanup method with the @PreDestroy annotation.
E. The developer should mark the cleanup method with the @PostDestroy annotation.
F. The cleanup method is executed in the transaction and security context of the last business method

Invocation.

Correct Answer: BCD
Section: (none)
Explanation

Explanation/Reference:

QUESTION 65
A developer creates a stateless session bean. This session bean needs data from a remote system. Reading this data takes a long time. Assume that the data will NOT change during the life time of the bean and that the information to connect to the remote system is defined in JNDI.

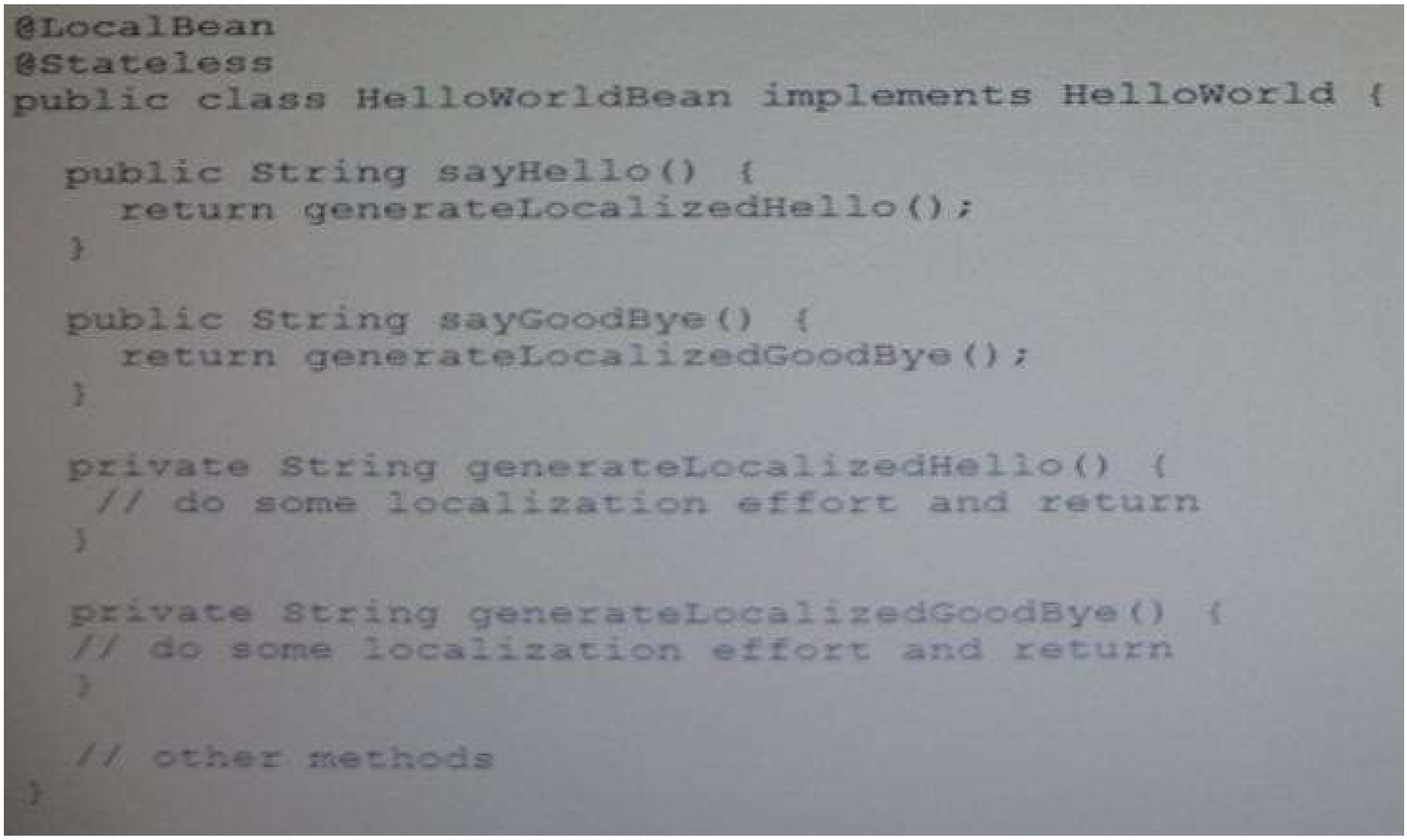
Which statement describes how to manage the data correctly?

A. Read the data in the bean's constructor.
B. The data can only be read in the bean's business methods.
C. Read the data in a method which is annotated with @PrePassivate.
D. Read the data in a method which is annotated with @PostActivate.
E. Read the data in a method which is annotated with @PostConstruct.

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 66
Suppose an EJB named HelloWorldBean is deployed as a standalone ejb-jar. Assuming the HelloWorldBean is implemented as follows:



```
@LocalBean
@Stateless
public class HelloWorldBean implements HelloWorld {

  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
}
```

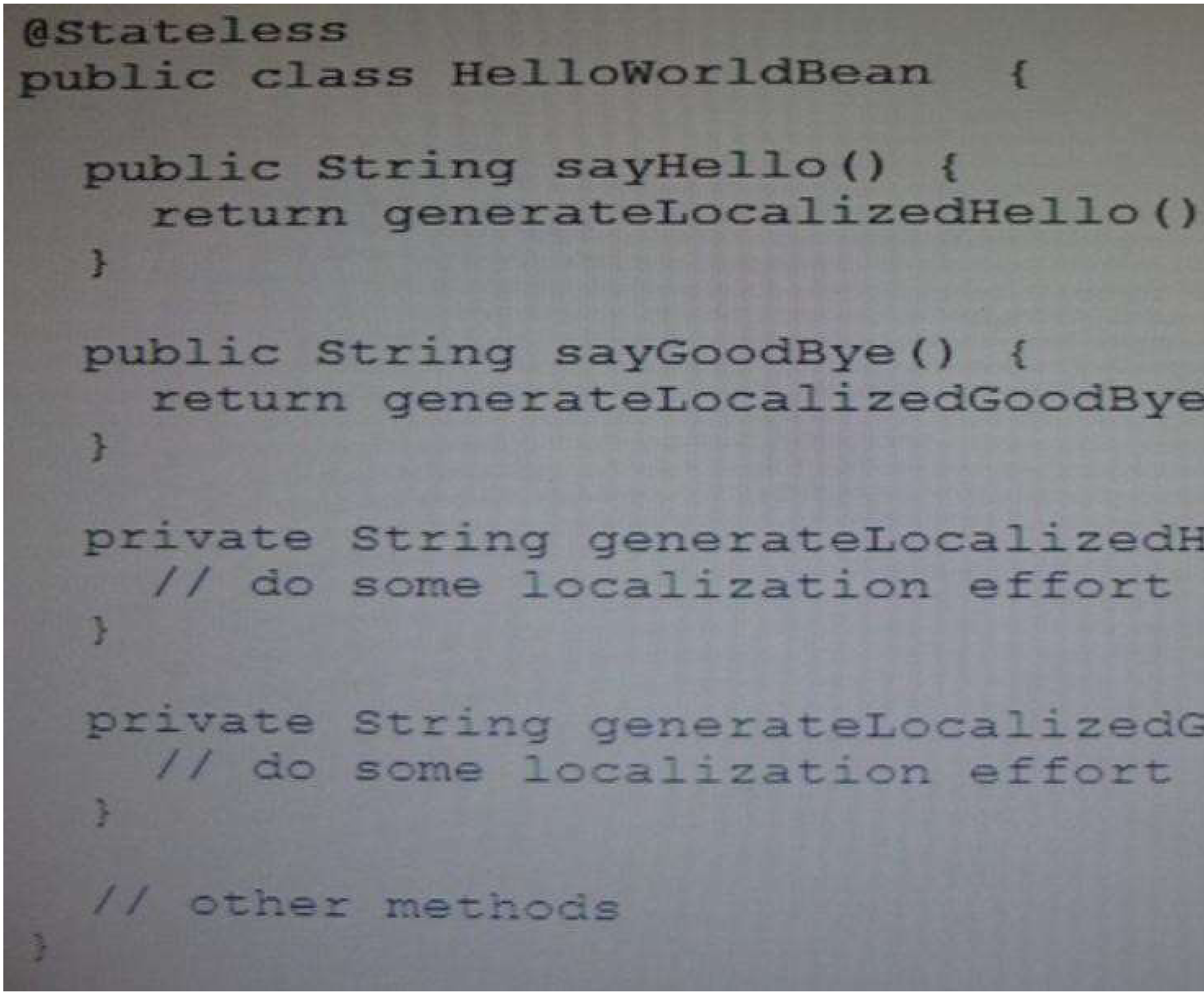
Which HelloWorldBean methods are accessible by another EJB within the same ejb-jar?

A. All of the methods declared in HelloWorldBean
B. All of the methods declared in HelloWorld and HelloWorldBean
C. All of the public methods declared in HelloWorldBean
D. All of the public methods declared in HelloWorld and all of the methods declared in HelloWorldBean

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
From JSR318 Enterprise JavaBeans 3.1, Final Release4.9.6 Business MethodsThe business method must be declared as public.
4.9.7 Session Bean's Business InterfaceIf the bean does not expose any other business interfaces (Local, Remote) or No-Interface view, and the bean class implements a single interface, that interface is assumed to be the business interface of the bean. This business interface will be a local interface unless the interface is designated as a remote business interface by use of the Remote annotation on the bean class or interface or by means of the deployment descriptor. 4.9.8 Session Bean's No-Interface ViewIf the bean exposes at least one other client view, the bean designates that it exposes ano-interface view by means of the @LocalBean annotation on the bean class or inthe deployment descriptor.All public methods of the bean class and any superclasses are exposed as business methodsthrough the no-interface view

QUESTION 67
Given the following stateless session bean:



```
@Stateless
public class HelloWorldBean   {

   public String sayHello() {
      return generateLocalizedHello()
   }

   public String sayGoodBye() {
      return generateLocalizedGoodBye
   }

   private String generateLocalizedH
      // do some localization effort
   }

   private String generateLocalizedG
      // do some localization effort
   }

  // other methods
}
```

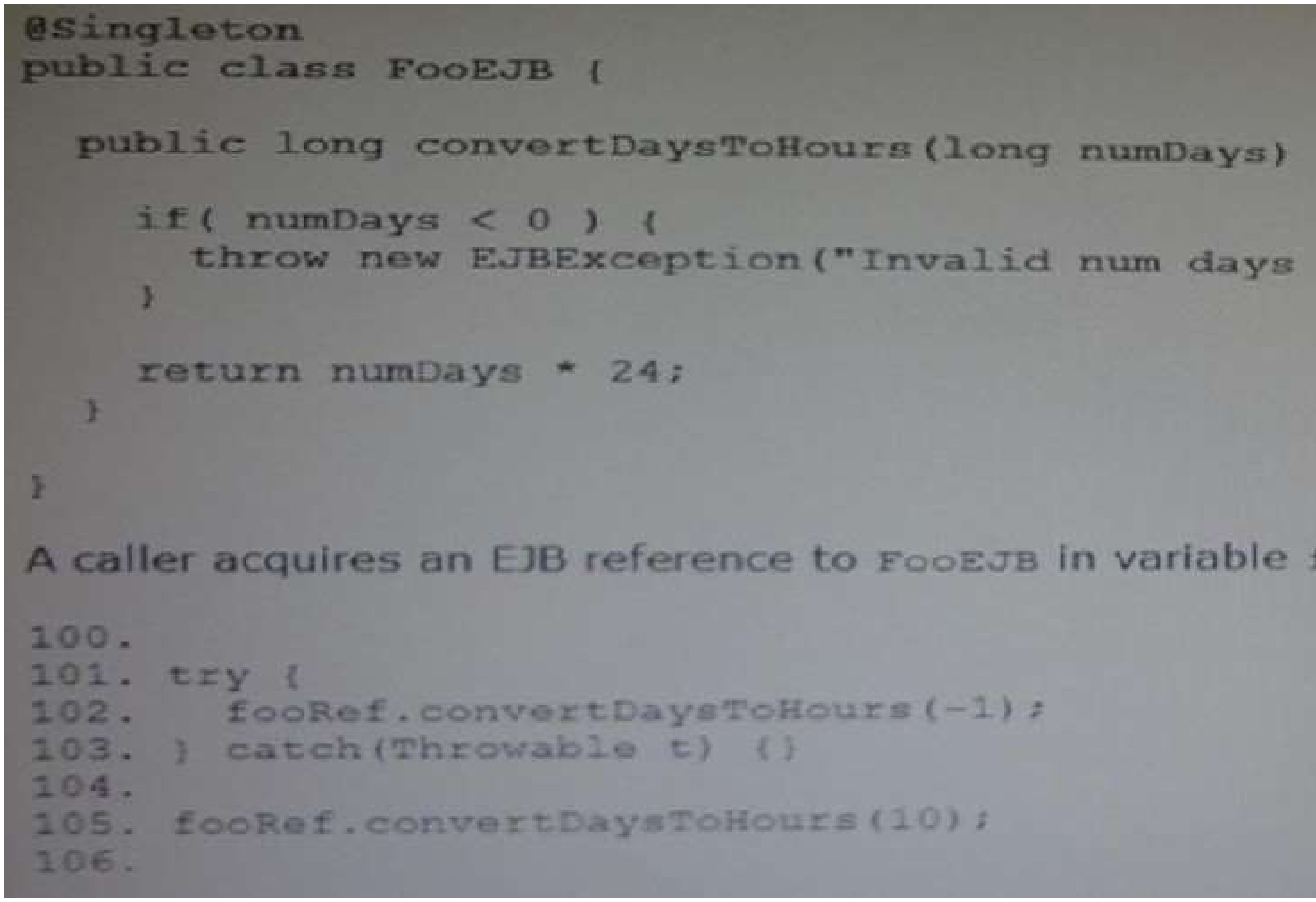
How would you change the EJB to prevent multiple clients from simultaneously accessing the sayHello method of a single bean instance?

A. Convert sayHello into a synchronized method
B. Execute the call to generateLocalizedHello in a synchronized block
C. Convert generateLocalizehello into a synchronized method
D. Convert HelloWordBean into a singleton bean
E. No changes are needed

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 68
Given singleton bean FooEJB:



```
@Singleton
public class FooEJB {

  public long convertDaysToHours(long numDays)

     if( numDays < 0 ) {
       throw new EJBException("Invalid num days
     }

     return numDays * 24;
  }

}
```

A caller acquires an EJB reference to FooEJB in variable

```
100.
101. try {
102.     fooRef.convertDaysToHours(-1);
103. } catch(Throwable t) {}
104.
105. fooRef.convertDaysToHours(10);
106.
```

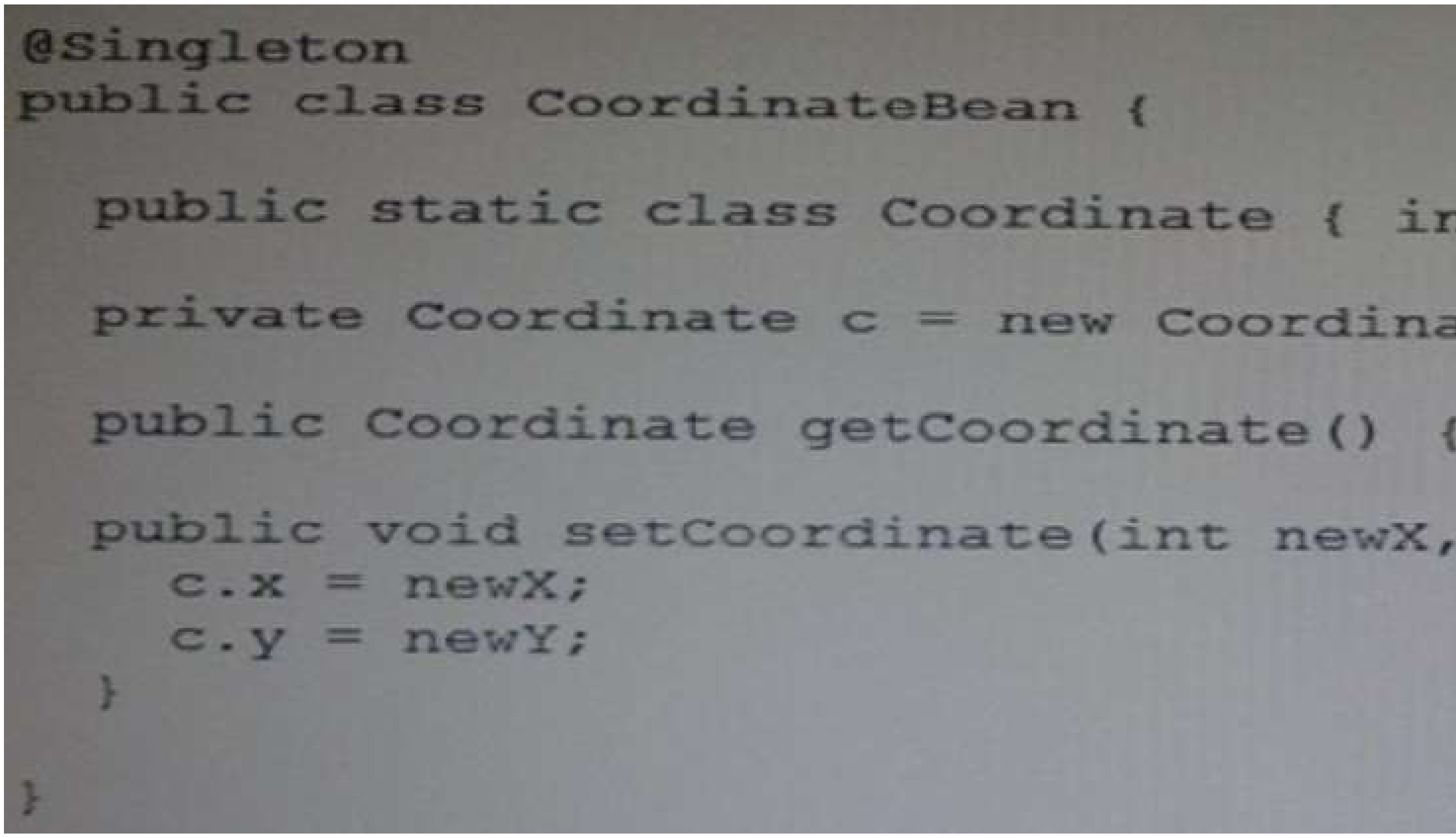
How many distinct FooEJB bean instances will be used to process the code on the lines 101-105?

A. 0
B. 1
C. 2

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Java has several design patterns Singleton Pattern being the most commonly used. Java Singleton pattern belongs to the family of design patterns, that govern the instantiation process. This design pattern proposes that at any time there can only be one instance of a singleton (object) created by the JVM.

QUESTION 69
A developer writes a Singleton bean that holds state for a single coordinate:



```
@Singleton
public class CoordinateBean {

  public static class Coordinate { in

  private Coordinate c = new Coordina

  public Coordinate getCoordinate() {

  public void setCoordinate(int newX,
    c.x = newX;
    c.y = newY;
  }

}
```

An update thread acquires an EJB reference to CoordinateBean and alternates between invoking SetCoordinate (0, 0) and SetCoordinate (1, 1) in a loop.

At the same time, ten reader threads each acquire an EJB reference to CoordinateBean and invoke getCoordinate () in a loop.

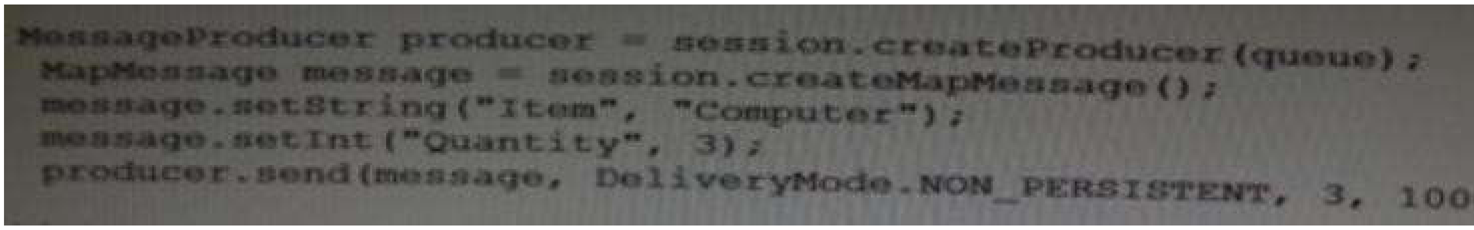
Which represents the set of all possible coordinate values [X, Y] returned to the reader threads?

A. [0, 0]
B. [1, 1]
C. [0, 0], [1, 1]
D. [0, 0], [0, 1], [1, 0], [1, 1]

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Accurate.

QUESTION 70
You are writing a client that sends a message to a JMS queue. The client includes the following code snippet:



```
MessageProducer producer = session.createProducer(queue);
 MapMessage message = session.createMapMessage();
 message.setString("Item", "Computer");
 message.setInt("Quantity", 3);
 producer.send(message, DeliveryMode.NON_PERSISTENT, 3, 100
```

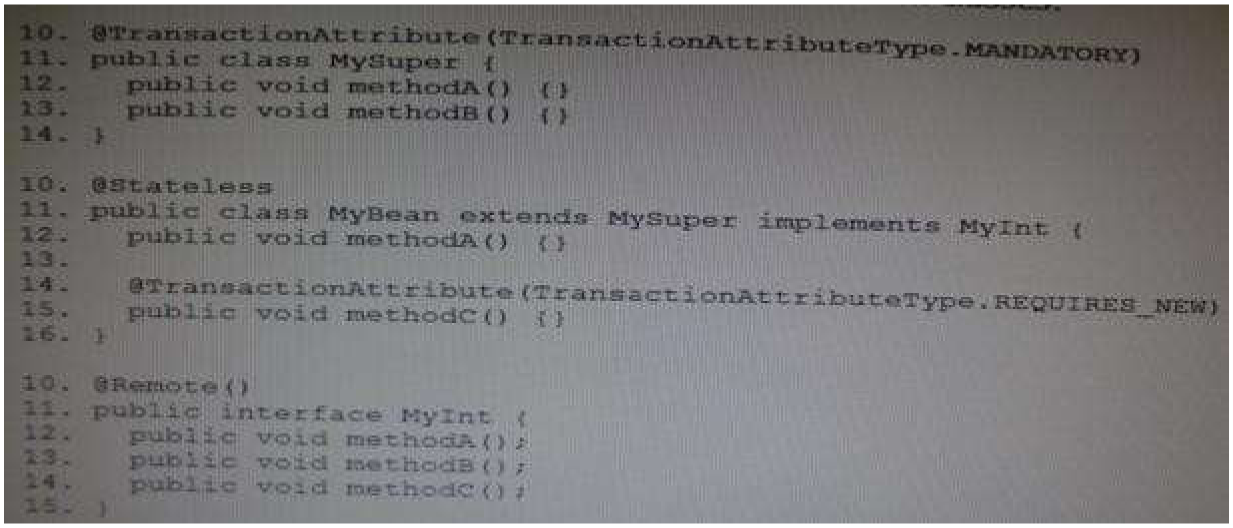
A. The message can be consumed by durable subscriber.
B. In the first line, the argument to the createProducer method needs to be cast to a Destination.
C. The message is sent using non-default deliver mode, priority, and expiration time values.
D. You could add more name-value pairs to the message body of the MapMessage if they were required by your application.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
createMapMessage

MapMessage createMapMessage()
throws JMSException
Creates a MapMessage object. A MapMessage object is used to send a self-defining set of name-value pairs, where names are String objects and values are primitive values in the Java programming language.

QUESTION 71
Given the following stateless session bean implementation classes:



```
@TransactionAttribute(TransactionAttributeType.MANDATORY)
public class MySuper {
  public void methodA() {}
  public void methodB() {}
}

@Stateless
public class MyBean extends MySuper implements MyInt {
  public void methodA() {}

  @TransactionAttribute(TransactionAttributeType.REQUIRES_NEW)
  public void methodC() {}
}

@Remote()
public interface MyInt {
  public void methodA();
  public void methodB();
  public void methodC();
}
```

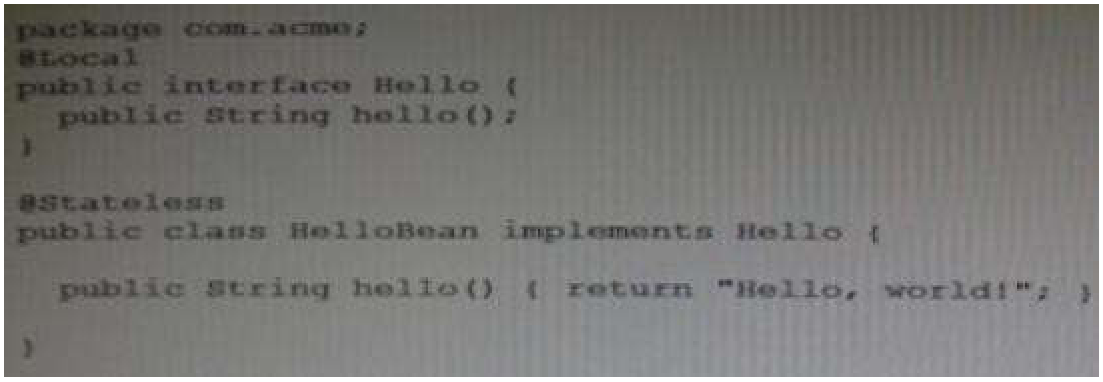
Assuming no other transaction-related metadata, what are the transaction attributes of methodA, methodB, and methodC respectively?

A. MANDATORY, MANDATORY , and MANDATORY
B. REQUIRED, MANDATORY, and REQUIRES_NEW
C. MANDATORY, MANDATORY , and REQUIRES_NEW
D. REQUIRED, REQUIRES_NEW, and REQUIRES_NEW

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:

QUESTION 72
A developer writes a stateless session bean HelloBean that exposes a local business interface Hello:



```
package com.acme;
@Local
public interface Hello {
  public String hello();
}

@Stateless
public class HelloBean implements Hello {

  public String hello() { return "Hello, world!"; }

}
```

The developer wants to test HelloBean using the EJB 3.1 Embeddable API. Given a main method with the following code:

100. EJBContainer container = EJBContainer.createEJBContainer();

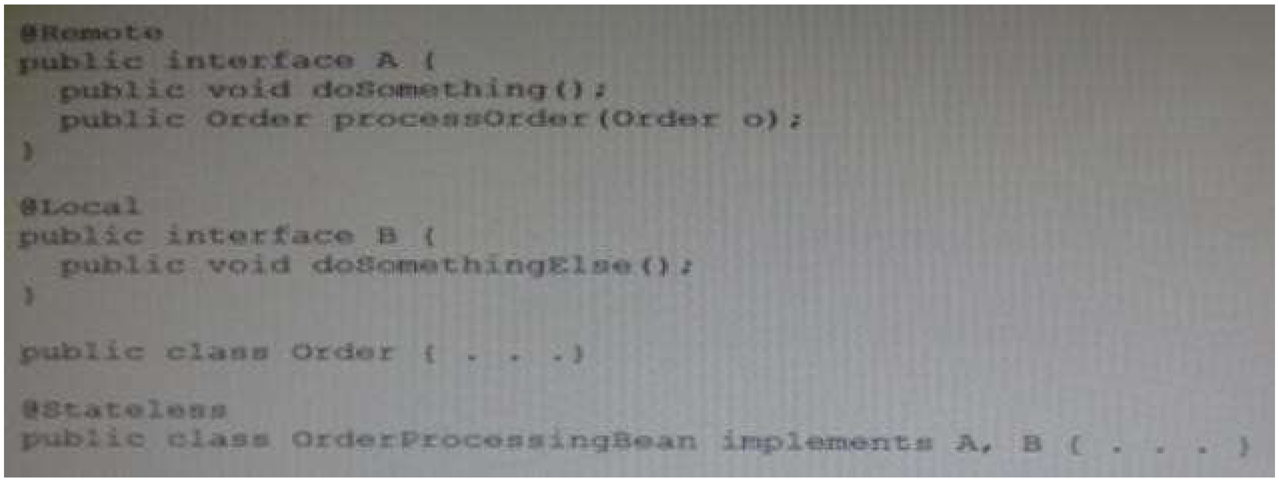
Assuming HelloBean is packaged in hello.jar and included in the classpath, which code correctly acquires a reference to HelloBean?

A. InitialContext ic = new InitialContext();Hello h = (Hello) ic.lookup("java:global/hello/HelloBean");
B. InitialContext ic = new InitialContext();Hello h = (Hello) ic.lookup("com.acme.Hello");
C. Context c = container.getContext();Hello h = (Hello) ic.lookup("com.acme.Hello");
D. Context c = container.getContext();Hello h = (Hello) ic.lookup("java:global/hello/HelloBean");

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

QUESTION 73
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

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
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.

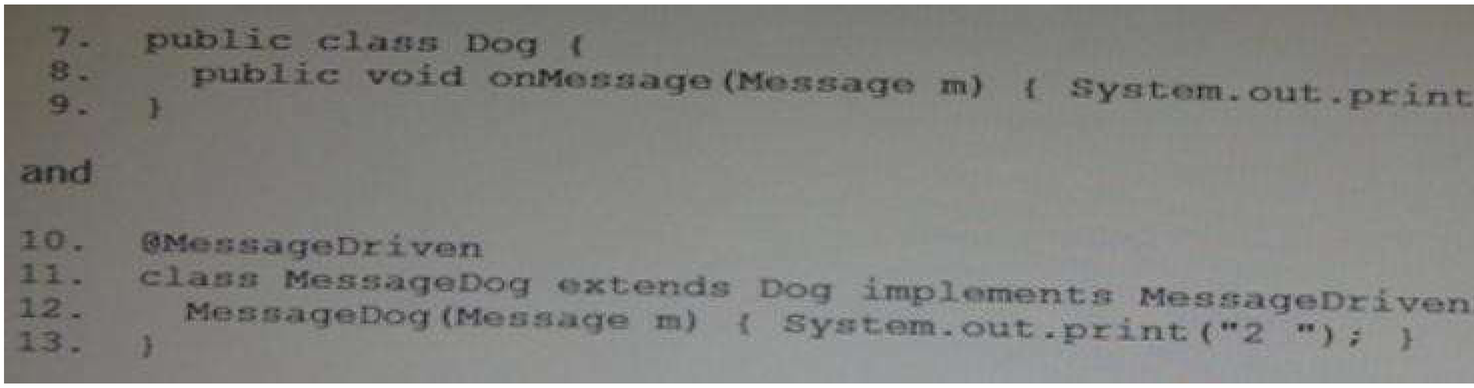
QUESTION 74
Which statement about message-driven beans is correct?

A. Each message-driven bean instance will be invoked by only one thread at a time.
B. When dispatching messages to message beam instances the container must preserve the order in which messages arrive.
C. If a message-driven bean is associated with a JMS queue, each bean instance in the pool will receive each message sent to the queue.
D. If a message driven bean is associated with a JMS durable subscription, each bean instance in the pool will receive each message sent to the durable subscription.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 75
Given code snippets from two files:



```
public class Dog {
  public void onMessage(Message m) { System.out.print
}
```

and

```
@MessageDriven
class MessageDog extends Dog implements MessageDriven
  MessageDog(Message m) { System.out.print("2 "); }
}
```

Which four code changes, when used together, create a valid JMS message-driven bean? (Choose four)

A. Make class MessageDog public
B. Make the MessageDog constructor no-arg
C. Make the MessageDog constructor public
D. Move the onMessage method to class MessageDog.
E. Change MessageDog so that it is NOT a subclass of Dog.
F. Make class MessageDog implement MessageListner instead of MessageDrivenBean.

Correct Answer: ABCF
Section: (none)
Explanation

Explanation/Reference:

QUESTION 76
Which is a valid Postconstruct method in a message-driven bean class?

A. @PostConstructpublic boolean init() { return true; }
B. @PostConstructprivate static void init () {}
C. @PostConstructprivate void init ()
D. @PostConstructpublic static void init () {}

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 77
Which statement is correct about a Java EF client of a message driven bean?

A. The client can use JNDI to obtain a reference to a message destination.
B. The client can use JNDI to obtain a reference to a dependency injection.
C. The client can use JNDI to obtain a reference to a message-driven bean instance.
D. The client can use JNDI to look up a reference to the message-driven bean's home interface.

Correct Answer: A
Section: (none)

Explanation

Explanation/Reference:

QUESTION 78
You are writing an EE component that functions as a message producer. The message producer sends message to a JMS queue. The component environment defines a resource-ref of type javax.jms.ConnectionFactory with the same jms/ConnectionFactory.

Which will correctly obtain a connection factory for a queue?

A. Context context = new initialContext();Connectionfactory confac = (ConnectionFactory) Context.lookup ("java: comp/env/jms/ConnectionFactory");
B. InitialContext Context = new Context () ;QueueConnectionFactory confac = (QueueConnectionFactory) context.lookup("java: comp/env/jms/Myfactory");
C. @Resource ("ConnectionFactory") private static QueueConnectionFactory confac;
D. @Resource (loopkup = "jms/QueueConnectionFactory") private static ConnectionFactoryconfac;

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 79
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.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
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 80
A bean developer writes a stateless session bean FooEJB with the following asynchronous business method:

```
@Asynchronous
public Future<Integer> fooAsync () {
System.out.println ("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)

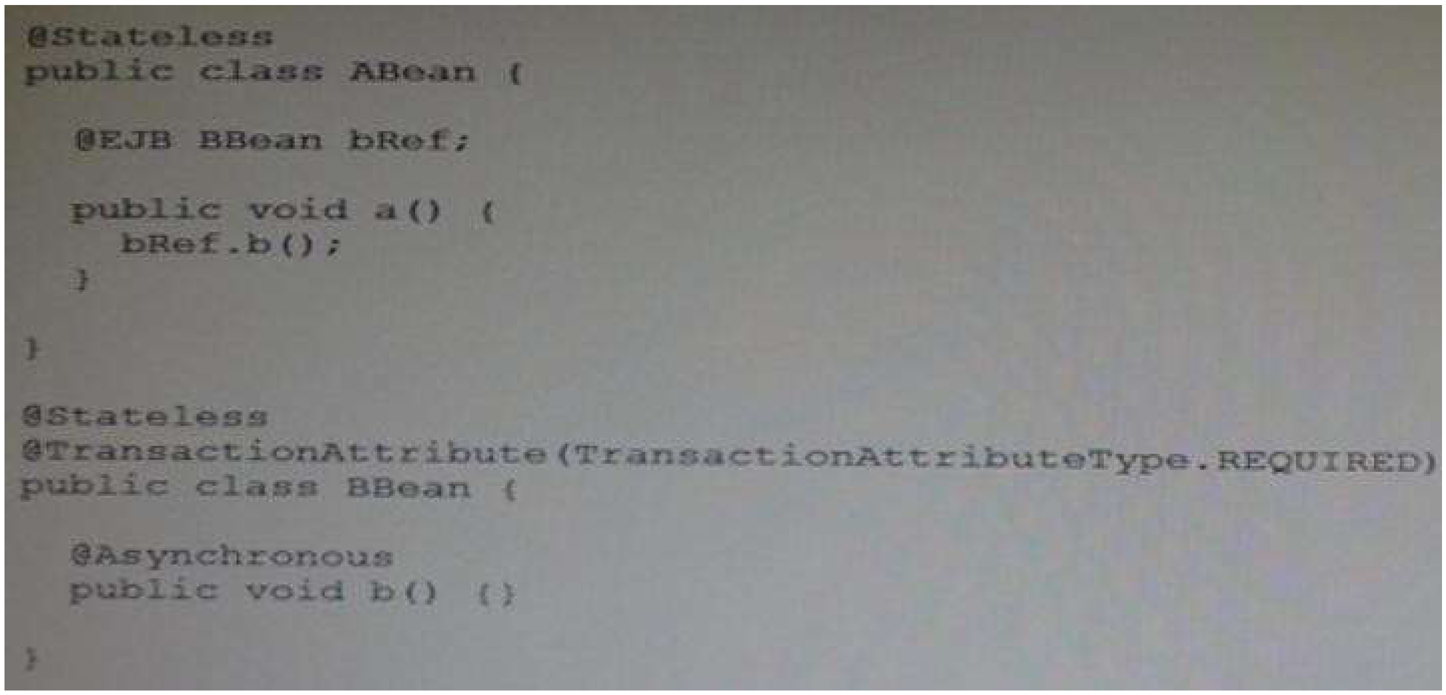
A. Begin end
B. Begin
C. End
D. 1
E. <no output>

Correct Answer: BE
Section: (none)
Explanation

Explanation/Reference:
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 81
Given two stateless session beans, ABean and BBean:



```
@Stateless
public class ABean {

   @EJB BBean bRef;

   public void a() {
      bRef.b();
   }

}

@Stateless
@TransactionAttribute(TransactionAttributeType.REQUIRED)
public class BBean {

   @Asynchronous
   public void b() {}

}
```

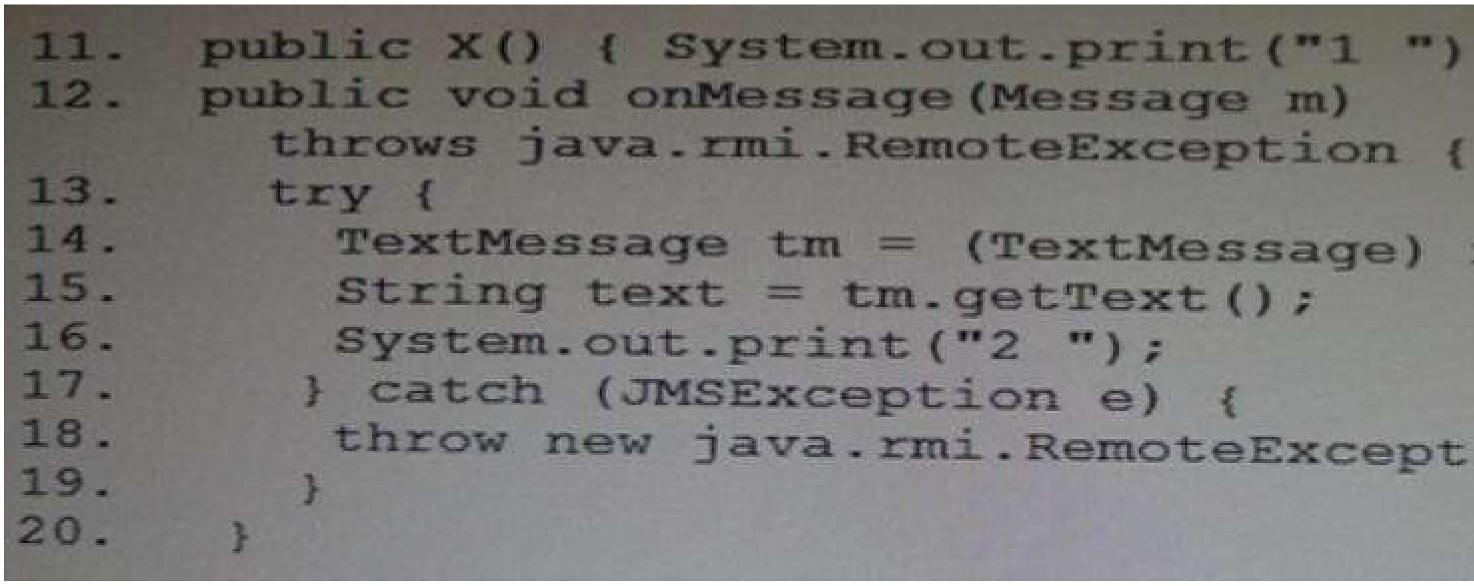
A client that is not executing within a transaction acquires an EJB reference to ABean and invokes the a() method on time. How many distinct transactions are started by the container after all processing has completed?

A. 0
B. 1
C. 2
D. 3

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Client transaction context does not propagate with an asynchronous method invocation. From the Bean Developer's view, there is never a transaction context flowing in from the client. This means, for example, that the semantics of the REQUIRED transaction attribute on an asynchronous method are exactly the same as REQUIRES_NEW.

QUESTION 82
Given this code snippet from a JMS message driven bean class X:



```
public X() { System.out.print("1 ")
public void onMessage(Message m)
   throws java.rmi.RemoteException {
   try {
     TextMessage tm = (TextMessage)
     String text = tm.getText();
     System.out.print("2 ");
    } catch (JMSException e) {
     throw new java.rmi.RemoteExcept
    }
 }
```

When this bean class handles a message, which is correct?

A. After a message delivery the result is 1.
B. After a message delivery the result is 2.
C. After a message delivery the result is 12.
D. After a message delivery an exception is thrown.
E. After a message delivery the result is unpredictable.
F. This is NOT a compliant JMS message-driven bean.

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:

QUESTION 83
Which two are true about the client view of a message-driven bean? (Choose two.)

A. References to message destinations can be infected.
B. References to message destinations cannot be looked up in the client's JNDI namespace.
C. Clients of a message destination need to know that the destination is listened to by a pool of message consumers,
D. Clients of a message destination do NOT need to know that the destination is listened to by a message-driven bean.

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:

QUESTION 84
Given the following stateful bean:

```
10. @Stateful
11. @TransactionAttribute(TransactionAttributeType.SUPPORTS)
12. public class VideoBean implements video {
13. / / insert code here
14. public void method () {}
15. }
```

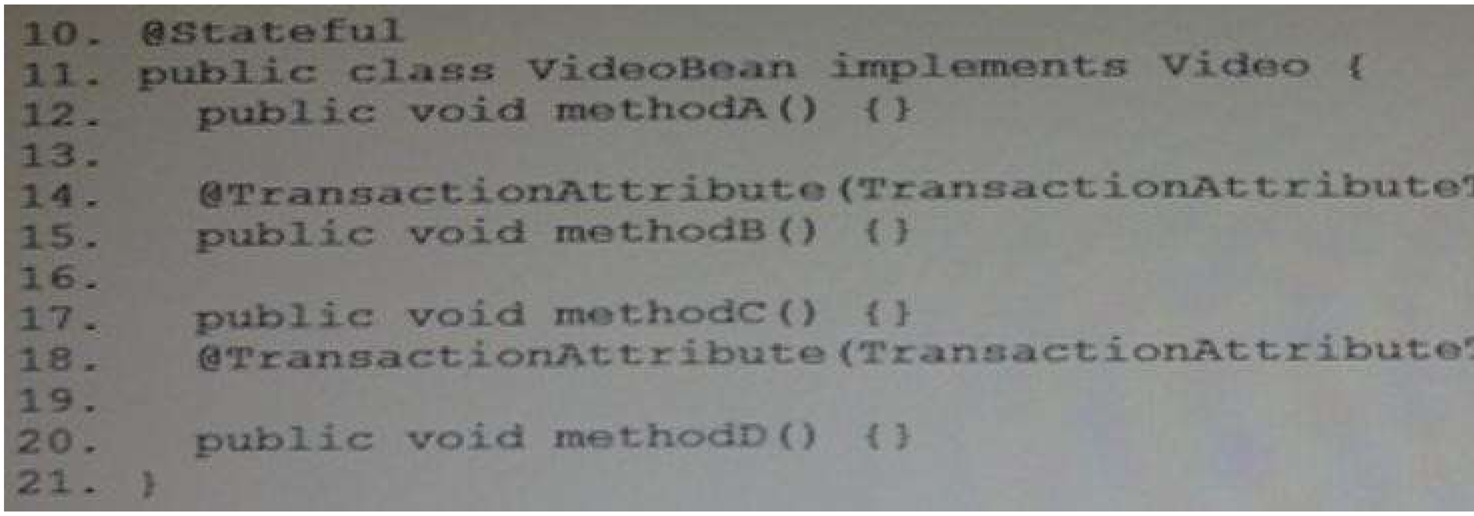
Assuming no other transaction-related metadata, which code can be added at line 13 to guarantee that business method methodA will execute only if invoked with an active transaction?

A. @TransactionAttribute ()
B. @transactionmanagement(TransactionAttributeType.CONTAINER)
C. @TransactionAttribute(transactionAttributeType.MANDATORY)
D. @transactionAttribute(TransactionattributeType.RECQUIRES_NEW)

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 85
Given the stateful session bean:



```
@Stateful
public class VideoBean implements Video {
   public void methodA() {}

   @TransactionAttribute(TransactionAttribute
   public void methodB() {}

   public void methodC() {}
   @TransactionAttribute(TransactionAttribute

   public void methodD() {}
}
```

Assuming no other transaction-related metadata, which is true?

A. methodB and methodC have transaction attribute SUPPORTS, which methodD has transaction attribute REQUIRED.
B. methodA and methodC have transaction attribute REQUIRES_NEW, while methodB has transaction attribute SUPPORTS.
C. methodC, methodD, and methodA have transaction attribute REQUIRED, and methodB has transaction attribute SUPPORTS.
D. methodB has transaction attribute SUPPORTS, methodD has transaction attribute REQUIRED, and methodA and methodC have transaction attribute REQUIRES_NEW.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 86
Which is true about caller security principal propagation for asynchronous EJB method Invocations?

A. Caller security principal propagates along with an asynchronous EJB method invocation.
B. Caller security principal does not propagate along with an asynchronous EJB method invocation.
C. Caller security principal propagates along with an asynchronous EJB method invocation only if the target bean has at least one protected method.
D. Caller security principal propagates along with an asynchronous EJB method invocation only if the calling bean has at least one protected method.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 87
Given a set of CMT bean methods with the following transaction attributes:

Method M1 = SUPPORTS
Method M2 = REQUIRED
Method M3 = NOT_SUPPORTED
Method M4 = REQUIRES NEW

And the following method invocation sequence:

Method Ml invokes Method M2
Method M2 invokes Method M3
Method M1 invokes Method M4
If Method Ml is invoked by a method that does NOT have a transaction context, which describes a possible scenario?

A. Method M1: no transactionMethod M2: new transaction Method M3: no transaction Method M4: new transaction
B. Method M1: no transactionMethod M2: Container throws EJBTransactionRequiredException
C. Method M1: new transactionMethod M2: runs in same transaction as M1Method M3: Container throws TransactionNotSupportedException
D. Method M1: no transactionMethod M2: new transactionMethod M3: Container throws TransactionNotSupportedException

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:

QUESTION 88
A stateful session bean needs to restore its conversational state to its initial state if the transaction in which the bean is participating rolls back.

Which bean method can be used to do this?

A. SessionContext.setRollbackOnly
B. SessionContext.getUserTransaction
C. SessionSynchronization.afterCompletion
D. SessionSynchrinization.beforeCompletion

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The afterCompletion method notifies a stateful session bean instance that a transaction commit protocol has completed, and tells the instance whether the transaction has been committed or rolled back.
Returns:
True if the current transaction is marked for rollback, false otherwise.

QUESTION 89
A developer implements a session bean which acts as a session facade for an application. This means that clients will only see this session bean's interface which offers the application interface, where are three distinct roles known at development time: "user", "admin", and "guest". The majority of the methods will be used by role "user". All methods must have role permissions active and roles may be added or changed in the future.

Which two scenarios are correct? (Choose two.)

A. The developer annotates the bean class with @PermitAll and annotates the methods used by role "guest" or "admin" individually.
B. The developer annotates the bean class with @DenyAll and annotates the methods used by role "user", "guest", or "admin" individually.
C. The developer defines individual method permissions for the methods used by roles "user "guest", and "admin" In the deployment descriptor.
D. The developer annotates the bean class with @RolesAllowed("user") and annotates the methods used by role "guest" or "admin" individually.
E. The developer defines a method permission with method name "*" and role "user" and adds individual method permissions for the methods used by roles "guest" and "admin" in the deployment descriptor.

Correct Answer: DE
Section: (none)
Explanation

Explanation/Reference:
D, E is perfect option.

QUESTION 90
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.

Correct Answer: BC
Section: (none)
Explanation

Explanation/Reference: