KillTest

Mejor calidad Mejor servicio

Examen

http://www.killtest.es

Renovación gratuita dentro de un año

Exam : **310-084**

Title : Sun Certified Web Component Developer for Java. EE 5 Upgrade

Version : Demo

1. Given the relationship:

The tag handler MyTag extends SimpleTagSupport. At runtime, the doTag method throws a SkipPageException.

Which three events occur after the SkipPageException is thrown? (Choose three.)

A. Evaluation of page2.jsp stops.

B. Evaluation of page1.jsp stops.

C. The MyTag instance is NOT reused.

D. Evaluation of page2.jsp continues.

E. Evaluation of page1.jsp continues.

Answer: ACE

2. A developer is designing a multi-tier web application and discovers a need to log each incoming client request.

Which two patterns, taken independently, provide a solution for this problem? (Choose two.)

A. Transfer Object

B. Service Locator

C. Front Controller

D. Intercepting Filter

E. Business Delegate

F. Model-View-Controller

Answer: CD

3. Which three are true about the HttpServletRequestWrapper class? (Choose three.)

A. The HttpServletRequestWrapper is an example of the Decorator pattern.

B. The HttpServletRequestWrapper can be used to extend the functionality of a servlet request.

C. A subclass of HttpServletRequestWrapper CANNOT modify the behavior of the getReader method.

D. An HttpServletRequestWrapper may be used only by a class implementing the javax.servlet.Filter interface.

E. An HttpServletRequestWrapper CANNOT be used on the request passed to the RequestDispatcher.include method.

F. An HttpServletRequestWrapper may modify the header of a request within an object implementing the javax.servlet.Filter interface.

Answer: ABF

4. Which two are valid values for the <transport-guarantee> element inside a <security-constraint> element of a web application deployment descriptor? (Choose two.)

A. NULL

B. SECURE

C. INTEGRAL

D. ENCRYPTED

E. CONFIDENTIAL

Answer: CD

5. Given a web application in which the request parameter productID contains a product identifier.

Which two EL expressions evaluate the value of the productID? (Choose two.)

A. ${productID}

B. ${param.productID}

C. ${params.productID}

D. ${params.productID[1]}

E. ${paramValues.productID}

F. ${paramValues.productID[0]}

G. ${pageContext.request.productID}

Answer: BF

6. Given the function invocation expression ${my:reverse("42")}, and that the function reverse is mapped into a Java method called reverse, which two are valid signatures for the Java method reverse? (Choose two.)

A. public int reverse(String val)

B. public String reverse(String val)

C. public static int reverse(String val)

D. public static String reverse(int val)

E. private static double reverse(double val)

F. public int reverse(String value, String name)

G. public static int reverse(int value, String name)

Answer: CD

7. For an HttpServletResponse response, which two create a custom header? (Choose two.)

A. response.setHeader("X-MyHeader", "34");

B. response.addHeader("X-MyHeader", "34");

C. response.setHeader(new HttpHeader("X-MyHeader", "34"));

D. response.addHeader(new HttpHeader("X-MyHeader", "34"));

E. response.addHeader(new ServletHeader("X-MyHeader", "34"));

F. response.setHeader(new ServletHeader("X-MyHeader", "34"));

Answer: AB

8. For a given ServletResponse response, which two retrieve an object for writing text data? (Choose two.)

A. response.getWriter()

B. response.getOutputStream()

C. response.getOutputWriter()

D. response.getWriter().getOutputStream()

E. response.getWriter(Writer.OUTPUT_TEXT)

Answer: AB

9. Given an HttpServletRequest request and HttpServletResponse response, which sets a cookie "username" with the value "joe" in a servlet?

A. request.addCookie("username", "joe")

B. request.setCookie("username", "joe")

C. response.addCookie("username", "joe")

D. request.addHeader(new Cookie("username", "joe"))

E. request.addCookie(new Cookie("username", "joe"))

F. response.addCookie(new Cookie("username", "joe"))

G. response.addHeader(new Cookie("username", "joe"))

Answer: F

10. Your company has a corporate policy that prohibits storing a customer's credit card number in any corporate database. However, users have complained that they do NOT want to re-enter their credit card number for each transaction. Your management has decided to use client-side cookies to record the user's credit card number for 120 days. Furthermore, they also want to protect this information during transit from the web browser to the web container; so the cookie must only be transmitted over HTTPS.

Which code snippet creates the "creditCard" cookie and adds it to the out going response to be stored on the user's web browser?

A. 10. Cookie c = new Cookie("creditCard", usersCard);
   11. c.setSecure(true);
   12. c.setAge(10368000);
   13. response.addCookie(c);

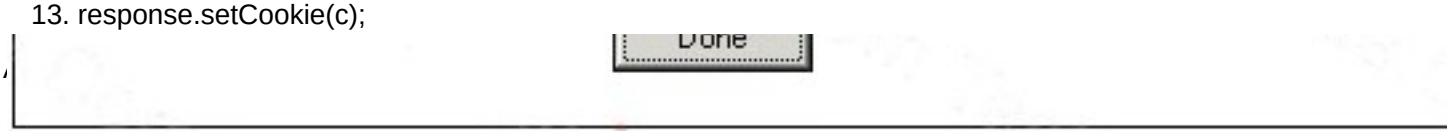
B. 10. Cookie c = new Cookie("creditCard", usersCard);

11. c.setHttps(true);
12. c.setMaxAge(10368000);
13. response.setCookie(c);

C. 10. Cookie c = new Cookie("creditCard", usersCard);
11. c.setSecure(true);
12. c.setMaxAge(10368000);
13. response.addCookie(c);

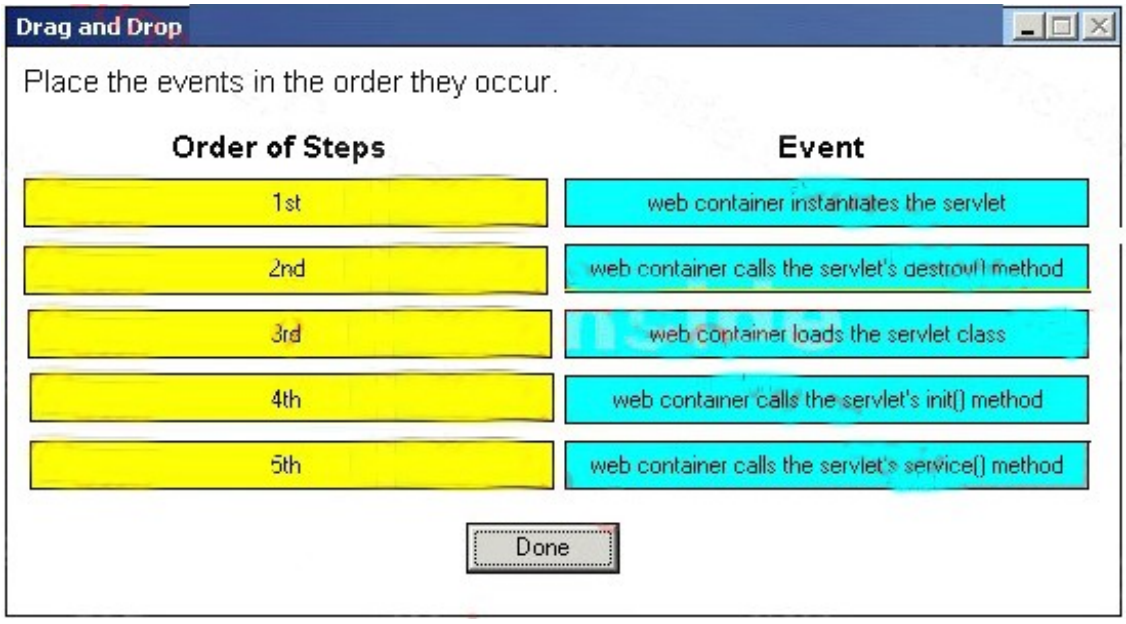
D. 10. Cookie c = new Cookie("creditCard", usersCard);
11. c.setHttps(true);
12. c.setAge(10368000);
13. response.addCookie(c);

E. 10. Cookie c = new Cookie("creditCard", usersCard);
11. c.setSecure(true);
12. c.setAge(10368000);
13. response.setCookie(c);



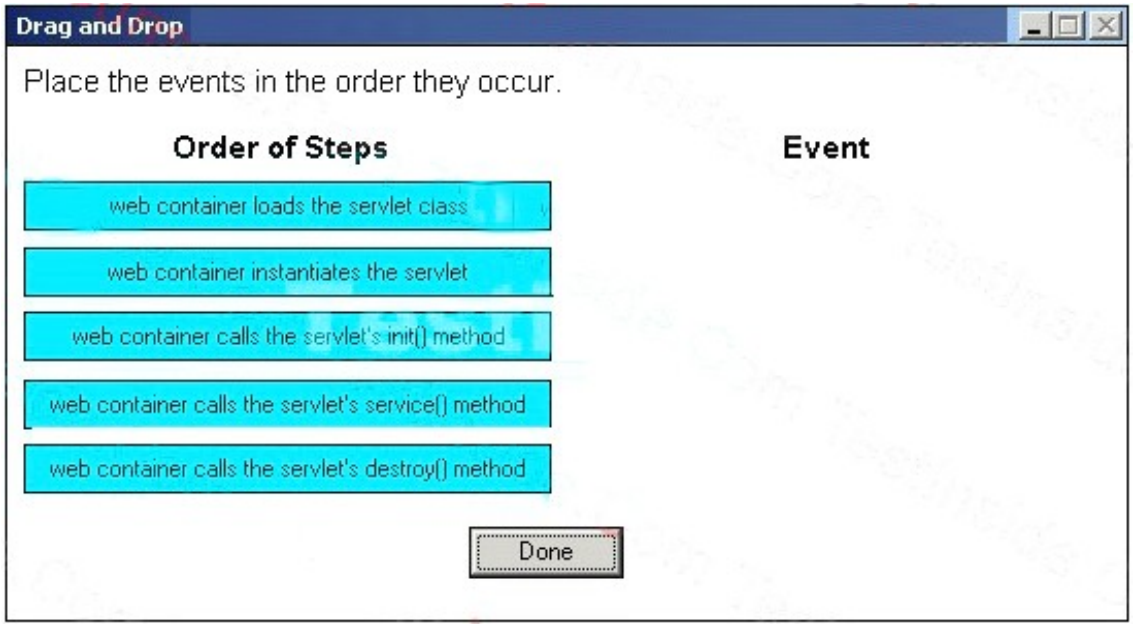
11. Click the Task button.

Place the events in the order they occur.



Answer:

Answer:



12. You are creating a servlet that generates stock market graphs. You want to provide the web browser with precise information about the amount of data being sent in the response stream.

Which two HttpServletResponse methods will you use to provide this information? (Choose two.)

A. response.setLength(numberOfBytes);

B. response.setContentLength(numberOfBytes);

C. response.setHeader("Length", numberOfBytes);

D. response.setIntHeader("Length", numberOfBytes);

E. response.setHeader("Content-Length", numberOfBytes);

F. response.setIntHeader("Content-Length", numberOfBytes);

Answer: BF

13. Which two prevent a servlet from handling requests? (Choose two.)

A. The servlet's init method returns a non-zero status.

B. The servlet's init method throws a ServletException.

C. The servlet's init method sets the ServletResponse's content length to 0.

D. The servlet's init method sets the ServletResponse's content type to null.

E. The servlet's init method does NOT return within a time period defined by the servlet container.

Answer: BE

14. Given an HttpSession session, a ServletRequest request, and a ServletContext context, which retrieves a URL to /WEB-INF/myconfig.xml within a web application?

A. session.getResource("/WEB-INF/myconfig.xml")

B. request.getResource("/WEB-INF/myconfig.xml")

C. context.getResource("/WEB-INF/myconfig.xml")

D. getClass().getResource("/WEB-INF/myconfig.xml")

Answer: C

15. Given:

```
11. public void service(ServletRequest request,
12.         ServletResponse response)
13.                 throws IOException {
14.     ServletContext ctx = getServletConfig().getServletContext();
15.     InputStream in =
16.         // insert code here
17. }
```

Which statement, at line 16, retrieves an InputStream for the file /WEB-INF/myresrc.bin?

A. new InputStream("/WEB-INF/myresrc.bin");

B. ctx.getInputStream("/WEB-INF/myresrc.bin");

C. ctx.getResourceAsStream("/WEB-INF/myresrc.bin");

D. new InputStream(new URL("/WEB-INF/myresrc.bin"));

E. getClass().getResourceAsStream("/WEB-INF/myresrc.bin");

Answer: C

16. Click the Exhibit button.

```
1. package com.example;
2.
3. import javax.servlet.http.*;
4.
5. public class MyWebDAV extends HttpServlet {
6. private String resourceDirectory;
7.
8. public MyWebDAV(String resDir) {
9. this.resourceDirectory = resDir;
10. }
11. public void doPut(HttpServletRequest req,
12. HttpServletResponse resp) {
13. // store file to resourceDirectory (code not shown)
20. }
21. public void doDelete(HttpServletRequest req,
22. HttpServletResponse resp) {
23. // remove file from resourceDirectory (code not shown)
30. }
31. }
```

As a maintenance feature, you have created this servlet to allow you to upload and remove files on your web server. Unfortunately, while testing this servlet, you try to upload a file using an HTTP request and on this servlet, the web container returns a 404 status.

What is wrong with this servlet?

A. HTTP does NOT support file upload operations.

B. The servlet constructor must NOT have any parameters.

C. The servlet needs a service method to dispatch the requests to the helper methods.

D. The doPut and doDelete methods do NOT map to the proper HTTP methods.

Answer: B

17. The tl:taskList and tl:task tags output a set of tasks to the response and are used as follows:

```
11. <tl:taskList>
12.     <tl:task name="Mow the lawn" />
13.     <tl:task name="Feed the dog" />
14.     <tl:task name="Do the laundry" />
15. </tl:taskList>
```

The tl:task tag supplies information about a single task while the tl:taskList tag does the final output. The tag handler for tl:taskList is TaskListTag. The tag handler for tl:task is TaskTag. Both tag handlers extend BodyTagSupport.

Which allows the tl:taskList tag to get the task names from its nested tl:task children?

A. It is impossible for a tag handler that extends BodyTagSupport to communicate with its parent and child tags.

B. In the TaskListTag.doStartTag method, call super.getChildTags() and iterate through the results. Cast each result to a TaskTag and call getName().

C. In the TaskListTag.doStartTag method, call getChildTags() on the PageContext and iterate through the results.

Cast each result to a TaskTag and call getName().

D. Create an addTaskName method in TaskListTag. Have the TaskListTag.doStartTag method, return BodyTag.EVAL_BODY_BUFFERED. In the TaskTag.doStartTag method, call super.getParent(), cast it to a TaskListTag, and call addTaskName().

E. Create an addTaskName method in TaskListTag. Have the TaskListTag.doStartTag method, return BodyTag.EVAL_BODY_BUFFERED. In the TaskTag.doStartTag method, call findAncestorWithClass() on the PageContext, passing TaskListTag as the class to find. Cast the result to TaskListTag and call addTaskName().

Answer: D

18. The sl:shoppingList and sl:item tags output a shopping list to the response and are used as follows:

```
11. <sl:shoppingList>
12.     <sl:item name="Bread" />
13.     <sl:item name="Milk" />
14.     <sl:item name="Eggs" />
15. </sl:shoppingList>
```

The tag handler for sl:shoppingList is ShoppingListTag and the tag handler for sl:item is ItemSimpleTag. ShoppingListTag extends BodyTagSupport and ItemSimpleTag extends SimpleTagSupport.

Which is true?

A. ItemSimpleTag can find the enclosing instance of ShoppingListTag by calling getParent() and casting the result to ShoppingListTag.

B. ShoppingListTag can find the child instances of ItemSimpleTag by calling super.getChildren() and casting each to an ItemSimpleTag.

C. It is impossible for ItemSimpleTag and ShoppingListTag to find each other in a tag hierarchy because one is a Simple tag and the other is a Classic tag.

D. ShoppingListTag can find the child instances of ItemSimpleTag by calling getChildren() on the PageContext and casting each to an ItemSimpleTag.

E. ItemSimpleTag can find the enclosing instance of ShoppingListTag by calling findAncestorWithClass() on the PageContext and casting the result to ShoppingListTag.

Answer: A

19. Given a JSP page:

```
11. <n:recurse>
12.     <n:recurse>
13.         <n:recurse>
14.             <n:recurse />
15.         </n:recurse>
16.     </n:recurse>
17. </n:recurse>
```

The tag handler for n:recurse extends SimpleTagSupport.

Assuming an n:recurse tag can either contain an empty body or another n:recurse tag, which strategy allows the tag handler for n:recurse to output the nesting depth of the deepest n:recurse tag?

A. It is impossible to determine the deepest nesting depth because it is impossible for tag handlers that extend SimpleTagSupport to communicate with their parent and child tags.

B. Create a private non-static attribute in the tag handler class called count of type int initialized to 0. Increment count in the doTag method. If the tag has a body, invoke the fragment for that body. Otherwise, output the value of count.

C. Start a counter at 1. Call getChildTags(). If it returns null, output the value of the counter. Otherwise, increment counter and continue from where getChildTags() is called. Skip processing of the body.

D. If the tag has a body, invoke the fragment for that body.Otherwise, start a counter at 1. Call getParent(). If it returns null, output the value of the counter Otherwise, increment the counter and continue from where getParent() is called.

Answer: D

20. You are creating a content management system (CMS) with a web application front-end. The JSP that displays a given document in the CMS has the following general structure:

```
1. <%-- tag declaration --%>
2. <t:document>
   ...
11. <t:paragraph>... <t:citation docID='xyz' /> ...</t:paragraph>
      ...
99. </t:document>
```

The citation tag must store information in the document tag for the document tag to generate a reference section at the end of the generated web page.

The document tag handler follows the Classic tag model and the citation tag handler follows the Simple tag model. Furthermore, the citation tag could also be embedded in other custom tags that could have either the Classic or Simple tag handler model.

Which tag handler method allows the citation tag to access the document tag?

A.
```
public void doTag() {
    JspTag docTag = findAncestorWithClass(this, DocumentTag.class);
  ((DocumentTag)docTag).addCitation(this.docID);
}
```

B. public void doStartTag() {
JspTag docTag = findAncestorWithClass(this, DocumentTag.class);
((DocumentTag)docTag).addCitation(this.docID);
}
C. public void doTag() {
Tag docTag = findAncestor(this, DocumentTag.class);
((DocumentTag)docTag).addCitation(this.docID);
}
D. public void doStartTag() {
Tag docTag = findAncestor(this, DocumentTag.class);
((DocumentTag)docTag).addCitation(this.docID);
}
Answer: A

21. Assume the tag handler for a st:simple tag extends SimpleTagSupport.
In what way can scriptlet code be used in the body of st:simple?
A. set the body content type to JSP in the TLD
B. Scriptlet code is NOT legal in the body of st:simple.
C. add scripting-enabled="true" to the start tag for the st:simple element
D. add a pass-through Classic tag with a body content type of JSP to the body of st:simple, and place the scriptlet code in the body of that tag
Answer: B

22. Under what two circumstances is the setJspBody method NOT called in a tag class that implements the SimpleTag interface? (Choose two.)
A. The tag is invoked without a body.
B. The doTag method throws an exception.
C. The <body-content> element has the value empty.
D. The tag is called with the attribute skip-body=true.
Answer: AC

23. You web application uses a lot of Java enumerated types in the domain model of the application. Built into each enum type is a method, getDisplay(), which returns a localized, user-oriented string. There are many uses for presenting enums within the web application, so your manager has asked you to create a custom tag that iterates over the set of enum values and processes the body of the tag once for each value; setting the value into a page-scoped attribute called, enumValue. Here is an example of how this tag is used:

```
11. <t:everyEnum type='com.example.Season'>
12.       <option value='${enumValue}'>${enumValue.display}</option>
13. </t:everyEnum>
14. </select>
```

You have decided to use the Simple tag model to create this tag handler.

Which tag handler method will accomplish this goal?

A.
```
public void doTag() throw JspException {
    try {
  for ( Enum value : getEnumValues() ) {
 pageContext.setAttribute("enumValue", value);
 getJspBody().invoke(getOut());
      }
    } (Exception e) { throw new JspException(e); }
 }
```

B.
```
public void doTag()
 throw JspException {
    try {
      for ( Enum value : getEnumValues() ) {
 getJspContext().setAttribute("enumValue", value);
 getJspBody().invoke(null);
      }
    } (Exception e) { throw new JspException(e); }
 }
```

C.
```
public void doTag() throw JspException {
    try {
      for ( Enum value : getEnumValues() ) {
 getJspContext().setAttribute("enumValue", value);
 getJspBody().invoke(getJspContext().getWriter());
      }
    }
 (Exception e) { throw new JspException(e); }
 }
```

D. public void doTag() throw JspException {

try {

for (

Enum value : getEnumValues()) {

pageContext.setAttribute("enumValue", value);

getJspBody().invoke(getJspContext().getWriter());

}

} (Exception e) { throw new JspException(e); }

}

Answer: B

24. Click the Exhibit button.

```
1. <?xml version="1.0" encoding="UTF-8" ?>
2.
3. <taglib xmlns="http://java.sun.com/xml/ns/j2ee"
4.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5.     xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee web-jsptaglibrary_2_0.xsd"
6.     version="2.0">
7.     <tlib-version>1.0</tlib-version>
8.     <short-name>h</short-name>
9.     <uri>http://example.com/tld/highlight</uri>
10.      <tag>
11.          <name>highlight</name>
12.          <tag-class>com.example.HighlightTag</tag-class>
13.          <body-content>scriptless</body-content>
14.          <attribute>
15.<name>color</name>
16.<required>true</required>
17.          </attribute>
18.          <dynamic-attributes>true</dynamic-attributes>
```

19. </tag>

20. </taglib>

The h:highlight tag renders its body, highlighting an arbitrary number of words, each of which is passed as an attribute (word1, word2, ...). For example, a JSP page can invoke the h:highlight tag as follows:

11. <h:highlight color="yellow" word1="high" word2="low">

12. high medium low

13. </h:highlight>

Given that HighlightTag extends SimpleTagSupport, which three steps are necessary to implement the tag handler for the highlight tag? (Choose three).

A. add a doTag method

B. add a doStartTag method

C. add a getter and setter for the color attribute

D. create and implement a TagExtraInfo class

E. implement the DynamicAttributes interface

F. add a getter and setter for the word1 and word2 attributes

Answer: ACE

25. You are building a web application that will be used throughout the European Union; therefore, it has significant internationalization requirements. You have been tasked to create a custom tag that generates a message using the java.text.MessageFormat class. The tag will take the resourceKey attribute and a variable number of argument attributes with the format, arg<N>. Here is an example use of this tag and its output:

<t:message resourceKey='diskFileMsg' arg0='MyDisk' arg1='1247' />

generates:

The disk "MyDisk" contains 1247 file(s).

Which Simple tag class definition accomplishes this goal of handling a variable number of tag attributes?

A. public class MessageTag extends SimpleTagSupport

implements VariableAttributes {

private Map attributes = new HashMap();

public void setVariableAttribute(String uri,

String name, Object value) {

this.attributes.put(name, value);

}

// more tag handler methods

```
}
```

B. The Simple tag model does NOT support a variable number of attributes.

C.

```
public class MessageTag extends
SimpleTagSupport
implements DynamicAttributes {
    private Map attributes = new HashMap();
    public void
putAttribute(String name, Object value) {
        this.attributes.put(name, value);
    }
    // more tag handler methods
}
```

D.

```
public class MessageTag extends SimpleTagSupport
implements VariableAttributes {
    private Map
attributes = new HashMap();
    public void putAttribute(String name, Object value) {
        this.attributes.put(name, value);
}
    // more tag handler methods
}
```

E.

```
public class MessageTag extends SimpleTagSupport
implements DynamicAttributes {
    private Map attributes = new HashMap();
    public void setDynamicAttribute(String uri, String name, Object value) {
        this.attributes.put(name, value);
    }
    // more tag handler methods
}
```

Answer: E

26. A developer is designing a web application that must verify for each request:

The originating request is from a trusted network.

The client has a valid session.

The client has been authenticated.

Which design pattern provides a solution in this situation?

A. Transfer Object

B. Session Facade

C. Intercepting Filter

D. Template Method

E. Model-View-Controller

Answer: C

27. A developer is designing a web application that makes many fine-grained remote data requests for each client request. During testing, the developer discovers that the volume of remote requests significantly degrades performance of the application.

Which design pattern provides a solution for this problem?

A. Flyweight

B. Transfer Object

C. Service Locator

D. Dispatcher View

E. Business Delegate

F. Model-View-Controller

Answer: B

28. A developer is designing the presentation tier for a web application that relies on a complex session bean. The session bean is still being developed and the APIs for it are NOT finalized. Any changes to the session bean API directly impacts the development of the presentation tier.

Which design pattern provides a means to manage the uncertainty in the API?

A. View Helper

B. Front Controller

C. Composite View

D. Intercepting Filter

E. Business Delegate

F. Chain of Responsibility

Answer: E

29. A developer is designing a web application which extensively uses EJBs and JMS. The developer finds that there is a lot of duplicated code to build the JNDI contexts to access the beans and queues. Further, because of the complexity, there are numerous errors in the code.

Which J2EE design pattern provides a solution for this problem?

A. Command

B. Transfer Object

C. Service Locator

D. Session Facade

E. Business Delegate

F. Data Access Object

Answer: C

30. A developer is designing the presentation tier for a web application which requires a centralized request handling to complete common processing required by each request.

Which design pattern provides a solution to this problem?

A. Remote Proxy

B. Front Controller

C. Service Activator

D. Intercepting Filter

E. Business Delegate

F. Data Access Object

Answer: B