Testpassport問題集

更に上のクオリティ　更に上のサービス

一年で無料進級することに提供する

Http://www.testpassport.jp

Exam : **1Z0-858J**

Title : Java Enterprise Edition 5 Web Component Developer Certified Professional Exam

Version : DEMO

1.XHTML と CSS のように求めて、Web 標準を使用して近代的なブラウザの機能を活用するために、Web アプリケーションは、単純な JSP ページからの JSP ドキュメント形式に変換されています。ただし、JSP は、/スクリプト/ screenFunctions.jsp の一つは、JavaScript ファイルのレートを生成します。このファイルは、画面固有の検証関数を作成するためにいくつかの Web フォームに含まれており、次のステートメントでこれらのページに含まれています：
10. <head>
11. <script src='/scripts/screenFunctions.jsp'
12. language='javascript'
13. type='application/javascript'> </script>
14. </head>
15. <!-- body of the web form -->
JSP のコードスニペットは、この JSP ドキュメントは JavaScript ファイルであることを宣言している？
A. <%@ page contentType='application/javascript' %>
B. <jsp:page contentType='application/javascript' />
C. <jsp:document contentType='application/javascript' />
D. <jsp:directive.page contentType='application/javascript' />
E. No declaration is needed because the web form XHTML page already declares the MIME type of the /scripts/screenFunctions.jsp file in the <script> tag.
答え: D

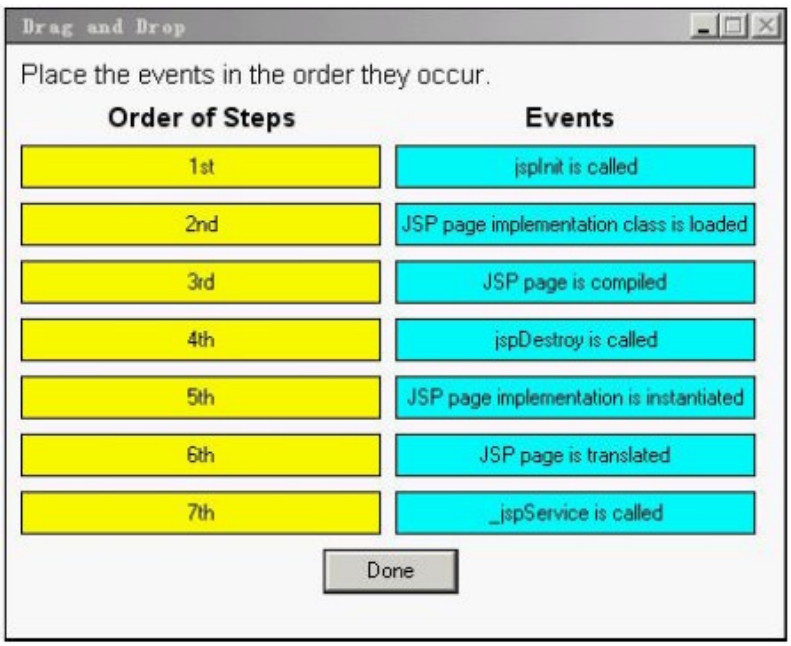
2.JSP のコードを与えられた：
10. <html>
11. <body>
12. <jsp:useBean id='customer' class='com.example.Customer' />
13. Hello, ${customer.title} ${customer.lastName}, welcome
14. to Squeaky Beans, Inc.
15. </body>
16. </html>
JSP コードのどの 3 種類が使用されています。？ （3 つ選択してください。）
A. Java code
B. template text
C. scripting code
D. standard action
E. expression language
答え: B,D,E

3.あなたの Web??アプリケーション用のカスタムタグのコレクションを構築しています。 /WEB-INF/myTags.xml：TLD ファイルは、ファイルに格納されています。あなたは、シンボリック名を使用して JSP でこれらのタグを参照して：
myTags、どのデプロイメント記述子要素は、このリンク betweenthe シンボリック名と TLD ファイルの名前を作るために使用する必要がありますか?
A. <taglib>
<name>myTags</name>
<location>/WEB-INF/myTags.xml</location>

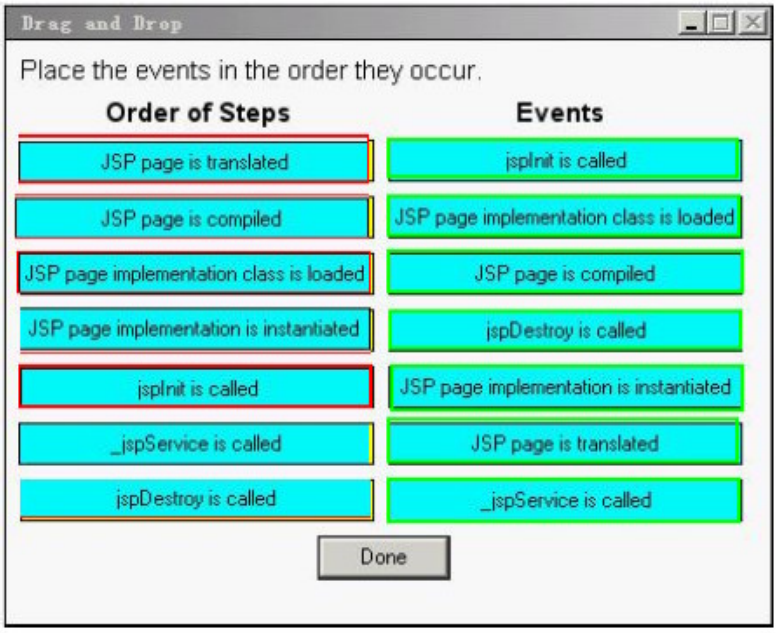
</taglib>
B. <tags>
<name>myTags</name>
<location>/WEB-INF/myTags.xml</location>
</tags>
C. <tags>
<tags-uri>myTags</taglib-uri>
<tags-location>/WEB-INF/myTags.xml</tags-location>
</tags>
D. <taglib>
<taglib-uri>myTags</taglib-uri>
<taglib-location>/WEB-INF/myTags.xml</taglib-location>
</taglib>
答え: D

4.どの暗黙オブジェクトは、デプロイメント ディスクリプタ内の<context-param>がエントリに関連付けられた値を取得するには、JSP ページで使用されていますか?
A. config
B. request
C. session
D. application
答え: D

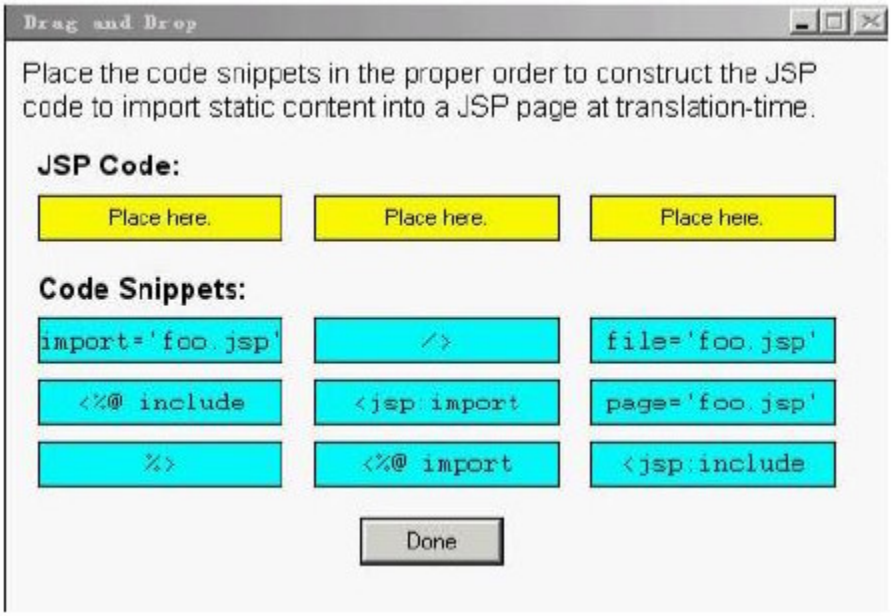
5.DRAG DROP
タスクボタンをクリックします。
彼らが発生した順序でイベントを配置します。



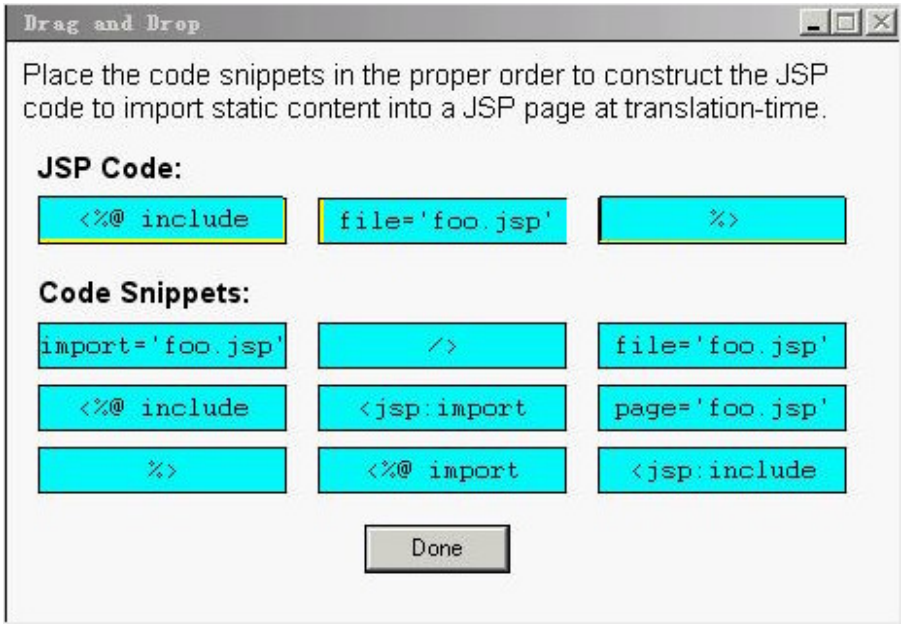
答え:



6.DRAG DROP
タスクボタンをクリックします。
変換時に JSP ページに静的コンテンツをインポートする JSP コードを構築するために適切な順序でコードスニペットを配置します。



答え:



7.あなたは、インスタンス変数とスクリプトレットのコードの多くを含む JSP を作成しました。残念なことに、大規模な負荷テストのコードの後に??、あなたの JSP 内でスクリプトレット、いくつかの競合状態を発見しました。これらの問題を解決するための重要な再コーディングを必要としていますが、予定より遅れて既にありました。どの JSP のコードスニペットは、これらの並行性の問題を解決するために使用できますか?

A. <%@ page isThreadSafe='false' %>
B. <%@ implements SingleThreadModel %>
C. <%! implements SingleThreadModel %>
D. <%@ page useSingleThreadModel='true' %>
E. <%@ page implements='SingleThreadModel' %>
答え: A

8.展示ボタンをクリックします。
属性"name"は"Foo"の値を持つ
このタグハンドラのタグが呼び出された場合、結果は何ですか?

```
 5. public class MyTagHandler extends
TagSupport {
 6.   public int doStartTag() throws
JspException {
 7.       try {
 8.         Writer out =
pageContext.getResponse().getWriter();
 9.         String name =
pageContext.findAttribute("name");
10.        out.print(name);
11.      } catch(Exception ex) { /* handle
exception */ }
12.      return SKIP_BODY;
13.   }
14
15.   public int doAfterBody() throws
JspException {
16.      try {
17.        Writer out =
pageContext.getResponse().getWriter();
18.        out.print("done");
19.      } catch(Exception ex) { /* handle
exception */ }
20.      return EVAL_PAGE;
21.   }
...
42. }
```

A. Foo
B. done
C. Foodone
D. 例外は、実行時にスローされます。
E. 出力はこのコードから生成されません。
F. コンパイルはこのために、コード内のエラーで失敗します。
答え: A

9.あなたはそれが重要な国際化の要件があり、従って、欧州連合全体で使用される Web アプリケーションを構築しています。あなたはレートの java.text.MessageFormat クラスを使用してメッセージを生成するカスタムタグを作成するために仕事を課されています。タグは、た ResourceKey 属性とフォーマットは、arg<N>を持つ引数の属性の数が可変になります。ここでは、このタグとその出力の使用例は、次のとおりです。
<t:message resourceKey='diskFileMsg' arg0='MyDisk' arg1='1247' />
generates: ディスク "MYDISK"は 1247 ファイル（複数可）が含まれています。
どのシンプル タグ クラス定義は、タグの属性の可変数を扱うのは、この目標を達成する？
A. パブリッククラスは、MessageTag SimpleTagSupport ををを拡張する
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. シンプル タグ モデルは、属性の変数の数をサポートしていません。
C. パブリッククラスは、MessageTag SimpleTagSupport をを拡張する
implements DynamicAttributes {
private Map attributes = new HashMap();
public void putAttribute(String name, Object value) {
this.attributes.put(name, value);}
// more tag handler methods }
D. パブリッククラスは、MessageTag SimpleTagSupport をを拡張する
implements VariableAttributes {
private Map attributes = new HashMap();
public void putAttribute(String name, Object value) {
this.attributes.put(name, value);}
// more tag handler methods}
E. パブリッククラスは、MessageTag SimpleTagSupport をを拡張する
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}
答え: E

10.JSP のコードを与えられ：
<% request.setAttribute("foo", "bar"); %>
従来のタグハンドラとコード：
5. public int doStartTag() throws JspException {
6. // insert code here
7. // return int
8. }
Web アプリケーション内の他の"foo"という属性が存在しないと仮定します。
6 行目に挿入 pageContext オブジェクト上のどの呼び出しでは、変数 x に"バー"を割り当てます。
A. String x = (String) pageContext.getAttribute("foo");
B. String x = (String) pageContext.getRequestScope("foo");
C.これは、doStartTag メソッド内から pageContext オブジェクトにアクセスすることはできません。
D. String x = (String)
pageContext.getRequest().getAttribute("foo");
E. String x = (String) pageContext.getAttribute("foo",
PageContext.ANY_SCOPE);
答え: D

11.どのタグファイルについての二つの文は本当ですか？ （2 つ選択してください。）
A. 従来のタグハンドラやタグファイルは、同じタグライブラリに存在することはできません。
B. / WEB-INF/tags/bar に位置 foo.tag という名前のファイルは、コンテナによって、タグ ファイルとして認識されています。

C. foo.tag という名前のファイルは、JAR ファイルにバンドルされているが TLD で定義されていない、コンテナの変換エラーをトリガします。
D. Web アプリケーションのルートディレクトリにある foo.tag という名前のファイルは、コンテナによって、タグ ファイルとして認識されています。
E. 両方のファイルが foo2.tag foo1.tag および/ WEB-INF/tags/bar に存在する場合、コンテナは、それらに同一のタグ ライブラリの一部を検討します。
答え: B,E

12.SL：買い物リストと SL：項目タグの出力応答に買い物リストとは、次のように使用されている：
11. <sl:shoppingList>
12 <sl:item name="Bread" />
13 <sl:item name="Milk" />
14 <sl:item name="Eggs" />
15. </sl:shoppingList>
SL のタグハンドラ：買い物リストやタグは、SL のハンドラを ShoppingListTag されています。項目が ItemSimpleTag されています。
ShoppingListTag BodyTagSupport は ItemSimpleTag SimpleTagSupport をを拡張し、拡張されています。これは本当ですか?
A. ItemSimpleTag は、getParent は（）を呼び出すと ShoppingListTag に結果をキャストすることによって ShoppingListTag のエンクロージングインスタンスを見つけることができます。
B. ShoppingListTag は ItemSimpleTag の super.getChildren（）を呼び出すと、各 ItemSimpleTag へキャストすることによって、子のインスタンスを見つけることができます。
C. 一つの単純なタグであり、他は古典的な日ですので、ItemSimpleTag とタグの階層でお互いを見つけるために ShoppingListTag ことは不可能です。
D. ShoppingListTag は、PageContext に ItemSimpleTag GetChildren を（）を呼び出すと ItemSimpleTag のそれぞれにキャストすることによって、子のインスタンスを見つけることができます。
E. ItemSimpleTag は、PageContext に findAncestorWithClass（）を呼び出すと ShoppingListTag に結果をキャストすることによって ShoppingListTag のエンクロージングインスタンスを見つけることができます。
答え: A

13.サーブレットはリクエストを受信していることに転送同じコンテナ内の別の Web の Web アプリケーション内のサーブレット B に。サーブレットは、サーブレット B とそのデータのアプリケーションで、他の Web サーブレットに見えてはいけませんとデータを共有する必要があります。れているオブジェクトは、B と共有し、データを格納することができます?
A. HttpSession
B. ServletConfig
C. ServletContext
D. HttpServletRequest
E. HttpServletResponse
答え: D

14.あなたのウェブサイトは、Web ページ上の例では、フォントや色の設定のために、多くのユーザーがカスタマイズできる機能を備えています。IT 部門は、すでに Java SE プラットフォームの API

lang.util.prefs パッケージを使用してユーザー設定のサブシステムを構築して、あなたは Web アプリケーションで、このサブシステムを再利用するために命じられている。あなた好みの工場を構築し、後で使用するためにアプリケーションスコープに格納するイベントリスナーを作成する必要があります。さらに、この工場は、データベースへの URL はこのようなデプロイメント記述子で宣言する必要がありますする必要があります。

42. <context-param>
43. <param-name>prefsDbURL</param-name>
44. <param-value>
45. jdbc:pointbase:server://dbhost:4747/prefsDB
46. </param-value>
47. </context-param>

どの部分のリスナクラスは、この目標を達成するのだろうか?

A. public class PrefsFactoryInitializer implements ContextListener {
public void contextInitialized(ServletContextEvent e) {
ServletContext ctx = e.getContext();
String prefsURL = ctx.getParameter("prefsDbURL");
PreferencesFactory myFactory = makeFactory(prefsURL);
ctx.putAttribute("myPrefsFactory", myFactory);}
// more code here}

B. public class PrefsFactoryInitializer implements ServletContextListener {
public void contextCreated(ServletContext ctx) {
String prefsURL = ctx.getInitParameter("prefsDbURL");
PreferencesFactory myFactory = makeFactory(prefsURL);
ctx.setAttribute("myPrefsFactory", myFactory);}
// more code here}

C. public class PrefsFactoryInitializer implements ServletContextListener {
public void contextInitialized(ServletContextEvent e) {
ServletContext ctx = e.getServletContext();
String prefsURL = ctx.getInitParameter("prefsDbURL");
PreferencesFactory myFactory = makeFactory(prefsURL);
ctx.setAttribute("myPrefsFactory", myFactory);
}
// more code here}

D. public class PrefsFactoryInitializer implements ContextListener
{
public void contextCreated(ServletContext ctx) {
String prefsURL = ctx.getParameter("prefsDbURL");
PreferencesFactory myFactory = makeFactory(prefsURL);
ctx.putAttribute("myPrefsFactory", myFactory);}
// more code here}

答え: C

15.開発者はアプリケーションがシャットダウンされようとしているときに通知する Web アプリケーションを望んでいる。どの 2 つのアクションは、この目標を達成するために必要ですか? （2 つ選択して

ください。）
A. JSP の page ディレクティブでリスナーを含める
B. 要素の<listener>を使用して TLD ファイル内でリスナーを構成する
C. Web アプリケーションのデプロイメント ディスクリプタ内の<servlet-destroy>要素を含んで
D. 要素の<listener>を使用して、アプリケーション デプロイメント ディスクリプタにリスナーを構成する
E. ServletContextListener サーブレットを Web アプリケーションのデプロイメントの実装の一部としてクラスを含む
F. 実装 ContextDestroyedListener Web アプリケーションの一部としてクラスを含む
展開
G. 実装すると、Web アプリケーションのデプロイメントの一部としてクラスを HttpSession リスナ属性を含める
答え: D,E

16.あなたの Web アプリケーションのフィルタを作成すると、フィルタは、javax.servlet.Filter を実装します。
どの二つのステートメントは真ですか？ （2 つ選択してください。）
A. フィルタクラスは、init メソッドと destroy メソッドを実装する必要があります。
B. フィルタクラスは、javax.servlet.FilterChain を実装する必要があります。
C. ときに次のフィルタにフィルタチェーンは、それがその doFilter メソッドで受信した同じ引数を渡す必要があります。
D. それは受信された実装したオブジェクトにフィルタを呼び出す方法
javax.servlet.FilterChain は、別のフィルタまたはサーブレットのいずれかを呼び出すことができます。
E. フィルタクラスには、他のものの間で取り、doFilter メソッドを実装する必要があります。
HttpServletRequest オブジェクトと HttpServletResponse オブジェクトに変換します。
答え: A,D

17.どの三人は HttpServletRequestWrapper クラスクラスについての真実ですか？ （3 つ選択してください。）
A. パターンは HttpServletRequestWrapper クラスの Decorator の例です。
B. HttpServletRequestWrapper クラスは、サーブレット リクエストの機能を拡張するために使用することができます。
C. HttpServletRequestWrapper クラスのサブクラスは、getReader メソッドの動作を変更することはできません。
D. インタフェースを実装するクラスは、javax.servlet.Filter で HttpServletRequestWrapper クラスにのみ使用することができます。
E. 方法 RequestDispatcher.include に渡された要求に応じて HttpServletRequestWrapper クラスには使用できません。
F. HttpServletRequestWrapper クラスでインターフェイスを実装する javax.servlet.Filter 内のオブジェクトへの要求のヘッダーを変更することができます。
答え: A,B,F

18.展示ボタンをクリックします。

```
// From file SourceServlet.java
11. public class SourceServlet extends
HttpServlet {
12.   public void service(HttpServletRequest
request,
13.                       HttpServletResponse
response)
14.         throws ServletException,
IOException {
15.     ServletContext
cxt=getServletConfig().getServletContext();
16.     RequestDispatcher rd =
17.               cxt.getRequestDispatcher("/dest
n");
18.     response.getWriter().println("hello
from source");
19.     response.flushBuffer();
20.     rd.forward(request, response);
21.   }
22. }

// From file DestinationServlet.java
11. public class DestinationServlet extends
HttpServlet {
12.   public void service(HttpServletRequest
request,
13.                       HttpServletResponse
response)
14.         throws ServletException,
IOException {
15.     response.getWriter().println("hello
from dest");
17.     response.flushBuffer();
18.   }
19. }
```

RequestDispatcher を要求されたリソースが利用可能で、DestinationServlet によって実装されています。
結果は何ですか?

A. 例外は SourceServlet によって実行時にスローされます。
B. 例外は DestinationServlet によって実行時にスローされます。
C. のみ"を dest からこんにちは"応答出力ストリームに表示されます。
D. どちらも "ソースからの hello"と "dest のからこんにちは"応答出力ストリームに表示されます。
答え: A

19.開発者は、同じブラウザインスタンスから、そのユーザから複数のリクエスト間で特定のユーザに関連付けられているすべてのサーブレットに使用可能な名前属性を、したいと考えています。
どの 2 つは、ハンドラタグ内からこの機能を提供する? （2 つ選択してください。）

A. pageContext.setAttribute("name", theValue);
B. pageContext.setAttribute("name", getSession());
C. pageContext.getRequest().setAttribute("name", theValue);
D. pageContext.getSession().setAttribute("name", theValue);
E. pageContext.setAttribute("name", theValue,
PageContext.PAGE_SCOPE);
F. pageContext.setAttribute("name", theValue,

PageContext.SESSION_SCOPE);
答え: D,F

20.MyServlet の定義を与えられた：
11. public class MyServlet extends HttpServlet {
12. public void service(HttpServletRequest request,
13. HttpServletResponse response)
14. throws ServletException, IOException {
15. HttpSession session = request.getSession();
16 session.setAttribute("myAttribute","myAttributeValue");
17. session.invalidate();
18. response.getWriter().println("value=" +
19. session.getAttribute("myAttribute"));
20. }
21 }
要求はの MyServlet に送信されます結果はどうなりますか?
A. IllegalStateException が実行時にスローされます。
B. InvalidSessionException で、実行時にスローされます。
C. 文字列 "値=0"は、応答ストリームに表示されます。
D. 文字列 "値= myAttributeValue"は、応答ストリームに表示されます。
答え: A