|  |  |  |  |
| --- | --- | --- | --- |
| **Final Exam** | | | |
|  |  |  |  |
|  | | | |
| 1. | What is "reflection" or "introspection" in JavaBeans? | | |
|  |  | (a) | It is the way a JavaBean uses getter and setter method names to deduce the attribute name. |
|  |  | (b) | It is a rule to create a default constructor in a JavaBean. |
|  |  | (c) | It is the way a Javabean inherits attributes from its parent class. |
|  |  | (d) | It is the way an exact copy of a JavaBean object is created. |
|  | | | |
| 2. | What is the JavaBean spec? | | |
|  |  | (a) | It is a set of rules that a JavaBean class needs to follow. |
|  |  | (b) | It is a set of rules on what classes a Javabean can inherit from. |
|  |  | (c) | It is set of rules on how to invoke methods in a JavaBean class. |
|  |  | (d) | It is a set of rules on how to overload methods in a Javabean class. |
|  | | | |
| 3. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | A JavaBean class need not inherit from any class. | |  | (ii) | A JavaBean class must be public. | |  | (iii) | Typically, a JavaBean's properties are manipulated via getters and setters. | |  | (iv) | You can overload methods in a Javabean. | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i), (ii) and (iii) only |
|  |  | (c) | (ii), (iii) and (iv) only |
|  |  | (d) | (i), (ii) and (iv) only |
|  | | | |
| 4. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | Every getter method in a JavaBean must have a corresponding class property. | |  | (ii) | Every property in a JavaBean must have a getter and setter. | |  | (iii) | Typically, a getter method in a JavaBean needs to begin with 'get' (or 'is' for boolean type attributes), followed by the name of the attribute with the first letter in upper case. | |  | (iv) | Typically, a setter method needs to begin with 'set', followed by the name of the attribute with the first letter in upper case. | | | |
|  |  | (a) | (i) and (ii) only |
|  |  | (b) | (ii), (iii) and (iv) only |
|  |  | (c) | (iii) and (iv) only |
|  |  | (d) | (i), (iii) and (iv) only |
|  | | | |
| 5. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | A custom tag class must implement (directly or indirectly via its parent) one of: Tag, IterationTag, and BodyTag interfaces. | |  | (ii) | TagSupport and BodyTagSupport classes implement the Tag and BodyTag interfaces respectively. | |  | (iii) | If your custom tag class inherits from TagSupport, you typically override the doStartTag() method to put your tag-specific processing. | |  | (iv) | BodyTag interface extends from the IterationTag interface, which in turn extends from the Tag interface. | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i), (ii) and (iv) only |
|  |  | (c) | (ii) and (iii) only |
|  |  | (d) | (i) and (iv) only |
|  | | | |
| 6. | Which of the sequence of calls below is valid? | | |
|  |  | (a) | doInitBody() -> doStartTag() -> doEndTag() -> doAfterBody() -> release() |
|  |  | (b) | doStartTag() -> doEndTag() -> doInitBody() -> doAfterBody() -> release() |
|  |  | (c) | doInitBody() -> doAfterBody() -> doStartTag() -> doEndTag() -> release() |
|  |  | (d) | doStartTag() -> doInitBody() -> doAfterBody() -> doEndTag() -> release() |
|  | | | |
| 7. | Which methods return an integer value that determines whether to execute methods downstream? | | |
|  |  | (a) | doStartTag(), doInitBody(), doAfterBody() and doEndTag() |
|  |  | (b) | doStartTag(), doAfterBody() and doEndTag() |
|  |  | (c) | doStartTag(), doInitBody() and doEndTag() |
|  |  | (d) | doInitBody(), doAfterBody() and doEndTag() |
|  | | | |
| 8. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | <rtexprvalue> in a custom tag defaults to true. | |  | (ii) | When an attribute's <rtexprvalue> in a custom tag is true, you can set that attribute using expression tags (<%= %>) in the JSP. | |  | (iii) | <required> in a custom tag defaults to true. | |  | (iv) | An attribute's <type> in a custom tag specifies the data type of that attribute (defaults to String). | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i) and (iii) only |
|  |  | (c) | (ii) and (iv) only |
|  |  | (d) | (ii), (iii) and (iv) only |
|  | | | |
| 9. | Which of the following are valid uses of Struts (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | Move the processing logic from the JSP to Java classes. | |  | (ii) | Have better control over the flow of calls amongst the JSP's. | |  | (iii) | Enforce a uniform, standard approach in the processing of requests by all developers. | |  | (iv) | Have better control over messages, labels used in the application thro Resource Bundles | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i) only |
|  |  | (c) | (i) and (iv) only |
|  |  | (d) | (ii) and (iii) only |
|  | | | |
| 10. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | Each action in the Struts configuration file must have an alias path to identify it in the list. | |  | (ii) | An action's alias path must be unique among all actions in the struts-config.xml file so that there is no ambiguity. | |  | (iii) | Any URL ending in .do is interpreted by the Web server to be a request for a Struts action. | |  | (iv) | The execute() method in an Action class returns an object of type ActionForward. | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (ii) and (iv) only |
|  |  | (c) | (i), (ii) and (iii) only |
|  |  | (d) | (i) and (iii) only |
|  | | | |
| 11. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | The technical name for 'web.xml' file is the "Deployment Descriptor". | |  | (ii) | A "forward" represents one possible outcome or pathway or resource (typically, a JSP) that is returned by the execute() method in the Action class. | |  | (iii) | You can name each forward whatever you want, as long as it is unique within each action. | |  | (iv) | The name of a forward (such as "success") serves as an alias name for the ActionForward object that represents the forward in question. | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (ii) and (iv) only |
|  |  | (c) | (i), (ii) and (iii) only |
|  |  | (d) | (i) and (iii) only |
|  | | | |
| 12. | The struts-config.xml file needs to be placed in: | | |
|  |  | (a) | WEB-INF/lib |
|  |  | (b) | WEB-INF/classes |
|  |  | (c) | Web app root directory |
|  |  | (d) | WEB-INF |
|  | | | |
| 13. | What does an execute() method in an Action class typically contain (select all that apply) ?   |  |  |  | | --- | --- | --- | |  | (i) | Back-end processing, i.e. doing the insert or update in a database | |  | (ii) | Validation and error checking | |  | (iii) | Populating derived attributes on an ActionForm | |  | (iv) | Populating the ActionForm object's attributes that map to <INPUT> elements on the HTML form | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i) and (iv) only |
|  |  | (c) | (i), (ii) and (iii) only |
|  |  | (d) | (ii), (iii) and (iv) only |
|  | | | |
| 14. | What is the reason you need to replace plain <INPUT> tags with Struts <html> tags in an HTML form? | | |
|  |  | (a) | To clean Java code from the JSP |
|  |  | (b) | To populate the attributes of the ActionForm with user-entered values |
|  |  | (c) | To make it look professional |
|  |  | (d) | To ensure that the form gets submitted as a POST |
|  | | | |
| 15. | Suppose you want to replace the following <INPUT> element:   |  | | --- | | <INPUT TYPE=TEXT NAME="firstName" SIZE=16/> |   in your HTML form with a Struts <html> equivalent.  Which of the following would work? | | |
|  |  | (a) | <html:input type="text" name="firstName" size = "16"/> |
|  |  | (b) | <html:input type="text" property="firstName" size = "16"/> |
|  |  | (c) | <html:text name="firstName" size = "16"/> |
|  |  | (d) | <html:text property="firstName" size = "16"/> |
|  | | | |
| 16. | Which of the following is true (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | <bean:define> can be used to create a scripting variable on the fly, whereas <bean:write> cannot. | |  | (ii) | <bean:write> does not have an 'id' attribute but <bean:define> does. | |  | (iii) | The 'name' attribute of <bean:write> is optional; it defaults to the ActionForm object for the current action | |  | (iv) | The 'id' attribute of <bean:define> is optional; it defaults to the ActionForm object for the current action | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (i), (ii) and (iv) only |
|  |  | (c) | (iii) and (iv) only |
|  |  | (d) | (i), (ii) and (iii) only |
|  | | | |
| 17. | Suppose you had the following content in ErrorMessages.properties file:   |  | | --- | | firstNameError=First name cannot be empty lastNameError=Last name cannot be empty |   and the following line in execute() method of your Action sub-class:   |  | | --- | | errors.add("firstName", new ActionMessage("firstNameError")); |   Which of the following would work, if you wanted the error message to show up next to the 'firstName' field on the HTML form (select all that apply)?   |  |  |  | | --- | --- | --- | |  | (i) | <html:errors property="firstName"/> | |  | (ii) | <html:errors property="firstNameError"/> | |  | (iii) | <html:messages property="firstName"/> | |  | (iv) | <html:messages property="firstNameError"/> | | | |
|  |  | (a) | All of the above |
|  |  | (b) | (ii) and (iv) only |
|  |  | (c) | (i) only |
|  |  | (d) | (i) and (iii) only |
|  | | | |
| 18. | What is a main disadvantage of using the <html:errors> tag? | | |
|  |  | (a) | It won't show any message if there are no errors |
|  |  | (b) | If there is more than one error, it will show the messages one after the other on the same line, so you need to use a <BR> tag in the error message itself to display messages on individual lines |
|  |  | (c) | You can't use <bean:write> to get the individual error messages |
|  |  | (d) | You can't use the <li> element to display individual messages |
|  | | | |
| 19. | Suppose you want to replace the use of <html:errors> with <html:messages> in your JSP. Which of the following would work? | | |
|  |  | (a) | |  | | --- | | <html:messages name="msg"> <li><bean:write id="msg"/></li> </html:messages> | |
|  |  | (b) | |  | | --- | | <html:messages id="msg"> <li><bean:write name="msg"/></li> </html:messages> | |
|  |  | (c) | |  | | --- | | <html:messages id="msg"> <li><bean:write property="msg"/></li> </html:messages> | |
|  |  | (d) | |  | | --- | | <html:messages name="msg"> <li><bean:write property="msg"/></li> </html:messages> | |
|  | | | |
| 20. | Suppose you are using <html:errors> element to show all errors at the top of the page. What change do you need to make in order to have the individual error messages show up next to an <INPUT> field? | | |
|  |  | (a) | Replace <html:errors> with <html:messages> |
|  |  | (b) | Move <html:errors> next to the field being validated, and set the 'property' attribute to contain the same key used to add the ActionMessage object to the errors. |
|  |  | (c) | Use an <html:messages> next to the field being validated, and set the 'id' attribute to contain the same key used to add the ActionMessage object to the errors. |
|  |  | (d) | Use a <bean:write> element next to the field being validated, and set the 'name' attribute to contain the same key used to add the ActionMessage object to the errors |
|  | | | |
| 21. | Suppose you have a <form-bean> as follows in struts-config.xml:   |  | | --- | | <form-beans> <form-bean name="NameForm" type="salesdept.NameActionForm" /> </form-beans> |   Also suppose that you want to validate the form in the execute() method of your Action sub-class. Which of the following <action> elements would you use to get it to work? | | |
|  |  | (a) | |  | | --- | | <action path="/inputname" name="NameForm" scope="request" forward="/InputName.jsp" validate="false"/>  <action path="/displayname" type="salesdept.DisplayNameAction" name="NameForm" scope="request" validate="true" input="/InputName.jsp"> <forward name="success" path="/DisplayName.jsp" /> </action> | |
|  |  | (b) | |  | | --- | | <action path="/inputname" name="NameForm" scope="request"> <forward name="success" path="/DisplayName.jsp" /> <forward name="inputError" path="/InputName.jsp" /> </action>  <action path="/displayname" type="salesdept.DisplayNameAction" name="NameForm" scope="request" forward="/InputName.jsp"/> | |
|  |  | (c) | |  | | --- | | <action path="/inputname" name="NameForm" scope="request" forward="/InputName.jsp" validate="true" input="/InputName.jsp"/>  <action path="/displayname" type="salesdept.DisplayNameAction" name="NameForm" scope="request"> <forward name="success" path="/DisplayName.jsp" /> </action> | |
|  |  | (d) | |  | | --- | | <action path="/inputname" name="NameForm" scope="request" forward="/InputName.jsp"/> <action path="/displayname" type="salesdept.DisplayNameAction" name="NameForm" scope="request"> <forward name="success" path="/DisplayName.jsp" /> <forward name="inputError" path="/InputName.jsp" /> </action> | |
|  | | | |
| 22. | In Question 21 above, assume that you only need to validate the last name, and check if it is not null or empty. Also assume ErrorMessages.properties is as shown in Question 17 above. Which of the following would be appropriate content for execute() method of DisplayNameAction.java? | | |
|  |  | (a) | |  | | --- | | public ActionForward execute (ActionMapping mapping, ActionForm actionForm, HttpServletRequest request, HttpServletResponse response ) { ActionMessages messages = new ActionMessages(); NameActionForm form = (NameActionForm)actionForm; String lastName = form.getLastName(); if (lastName == null || lastName.trim().equals("") ) { messages.add("lastName", new ActionMessage("lastNameError")); request.saveErrors(messages); return mapping.findForward("inputError"); } else return mapping.findForward("success"); } | |
|  |  | (b) | |  | | --- | | public ActionForward execute (ActionMapping mapping, ActionForm actionForm, HttpServletRequest request, HttpServletResponse response ) { ActionMessages messages = new ActionMessages(); NameActionForm form = (NameActionForm)actionForm; String lastName = form.getLastName(); if (lastName == null || lastName.trim().equals("") ) { messages.add("lastNameError", new ActionMessage("lastName")); request.saveErrors(messages); return mapping.findForward("inputError"); } else return mapping.findForward("success"); } | |
|  |  | (c) | |  | | --- | | public ActionForward execute (ActionMapping mapping, ActionForm actionForm, HttpServletRequest request, HttpServletResponse response ) { ActionMessages messages = new ActionMessages(); NameActionForm form = (NameActionForm)actionForm; String lastName = form.getLastName(); if (lastName == null || lastName.trim().equals("") ) { messages.add("lastName", new ActionMessage("lastNameError")); saveErrors(request, messages); return mapping.findForward("inputError"); } else return mapping.findForward("success"); } | |
|  |  | (d) | |  | | --- | | public ActionForward execute (ActionMapping mapping, ActionForm actionForm, HttpServletRequest request, HttpServletResponse response ) { ActionMessages messages = new ActionMessages(); NameActionForm form = (NameActionForm)actionForm; String lastName = form.getLastName(); if (lastName == null || lastName.trim().equals("") ) { messages.add("lastNameError", new ActionMessage("lastName")); saveErrors(request, messages); return mapping.findForward("inputError"); } else return mapping.findForward("success"); } | |
|  | | | |
| 23. | Assuming that you want to use the same ErrorMessages.properties file of Question 17 above, and want to perform error checking on just the last name via the validate() method of the ActionForm sub-class, which of the following would work? | | |
|  |  | (a) | |  | | --- | | ActionErrors errors = new ActionErrors(); if ( lastName == null || lastName.trim().equals("") ) { errors.add("lastName", new ActionError("lastNameError")); } return errors; | |
|  |  | (b) | |  | | --- | | ActionErrors errors = new ActionErrors(); if ( lastName == null || lastName.trim().equals("") ) { errors.add("lastName", new ActionMessage("lastNameError")); } return errors; | |
|  |  | (c) | |  | | --- | | ActionErrors errors = new ActionErrors(); if ( lastName == null || lastName.trim().equals("") ) { errors.add("lastNameError", new ActionError("lastName")); } return errors; | |
|  |  | (d) | |  | | --- | | ActionErrors errors = new ActionErrors(); if ( lastName == null || lastName.trim().equals("") ) { errors.add("lastNameError", new ActionMessage("lastName")); } return errors; | |
|  | | | |
| 24. | Suppose you want to test whether a "mode" attribute on the request is equal to "displayonly" or not, using <logic:equal> tag in your JSP. Which of the following would work in this case? | | |
|  |  | (a) | <logic:equal id="mode" value="displayonly"> |
|  |  | (b) | <logic:equal id="mode" property="displayonly"> |
|  |  | (c) | <logic:equal name="mode" value="displayonly"> |
|  |  | (d) | <logic:equal name="mode" property="displayonly"> |
|  | | | |
| 25. | Suppose you are using CustomerDBObject.java of Lesson 6 Chapter 3. Also suppose that "customerList" is an object of type List<CustomerDBObject> stored on the request that contains a List of CustomerDBObject objects. In addition, say that you have a <logic:iterate> tag in your JSP as follows:   |  | | --- | | <logic:iterate id="customer" name="customerList"> |   Finally suppose that for each row that the <logic:iterate> is iterating over, you want to use a Struts <html:link> in your JSP that has a destination URL as "/editcustomer.do", sending it the 'customerId' attribute of the CustomerDBObject.  Which of the following would be a valid <html:link> element that would work in this case? | | |
|  |  | (a) | |  | | --- | | <html:link page="/editcustomer.do" id="customerId" name="customer" property="custId"> Edit </html:link> | |
|  |  | (b) | |  | | --- | | <html:link page="/editcustomer.do" id="custId" name="customer" property="customerId"> Edit </html:link> | |
|  |  | (c) | |  | | --- | | <html:link page="/editcustomer.do" paramId="customerId" paramName="customer" paramProperty="custId"> Edit </html:link> | |
|  |  | (d) | |  | | --- | | <html:link page="/editcustomer.do" paramId="custId" paramName="customer" paramProperty="customerId"> Edit </html:link> | |