JAVA means DURGA SOFT

JAVA FAQ`S

Very important Questions in STRUTS

DURGA M.Tech
(Sun certified & Realtime Expert)
Ex. IBM Employee
Trained Lakhs of Students
for last 14 years across INDIA

India's No.1 Software Training Institute

DURGASOFT

www.durgasoft.com Ph: 9246212143 ,8096969696

STRUTS

Q 1. What is MVC?

Model-View-Controller (MVC) is a design pattern put together to help control change. MVC decouples interface from business logic and data.
Model: The model contains the core of the application's functionality. The model enca psulates the state of the application. Sometimes the only functionality it contains is state. It knows nothing about the view or controller.
View: The view provides the presentation of the model. It is the *look* of the application. The view can access the model getters, but it has no knowledge of the setters. In addition, it knows nothing about the controller. The view should be notified when changes to the model occur.
Controller: The controller reacts to the user input. It creates and sets the model.

Q 2. What is a framework?

Framework is made up of the set of classes which allow us to use a library in a best possible way for a specific requirement.

Q 3. What is Struts framework?

Struts framework is an open-source framework for developing the web applications in Java EE, based on MVC-2 architecture. It uses and extends the Java Servlet API. Struts is robust architecture and can be used for the development of application
of any size. Struts framework makes it much easier to design scalable, reliable Web applications with Java. Struts provides its own Controller component and integrates with other technologies to provide the Model and the View. For the Model, Struts can interact with standard data access technologies, like JDBC and EJB, as well as most any third-party packages, like Hibernate, iBATIS, or Object Relational Bridge. For the View, Struts works well with JavaServer Pages, including JSTL and JSF, as well as Velocity Templates, XSLT, and other presentation systems.

Q 4. What is Jakarta Struts Framework?

Jakarta Struts is open source implementation of MVC (Model-View-Controller) pattern for the development of web based applications. Jakarta Struts is robust architecture and can be used for the development of application of any size. Struts framework makes it much easier to design scalable, reliable Web applications with Java.

Q 5. What is ActionServlet?

The class org.apache.struts.action.ActionServlet is the called the ActionServlet. In the the Jakarta Struts Framework this class plays the role of controller. All the requests to the server goes through the controller. Controller is responsible for handling all the requests.

Q 6. What is role of ActionServlet?

ActionServlet performs the role of Controller:

- Process user requests
- Determine what the user is trying to achieve according to the request
- Pull data from the model (if necessary) to be given to the appropriate view,
- Select the proper view to respond to the user
- Delegates most of this grunt work to Action classes
- Is responsible for initialization and clean-up of resources

Q 7. What is Action Class?

Any java class which extends from org.apache.struts.action.Action is called Action class. The Action is part of the controller. The purpose of Action Class is to translate the HttpServletRequest to the business logic. To use the Action, we need to Subclass and overwrite the execute() method. The ActionServlet (commad) passes the parameterized class to Action Form using the execute() method. There should be no database interactions in the action. The action should receive the request, call business objects (which then handle database, or interface with J2EE, etc) and then determine where to go next. Even better, the business objects could be handed to the action at runtime (IoC style) thus removing any dependencies on the model. The return type of the execute method is ActionForward which is used by the Struts Framework to forward the request to the file as per the value of the returned ActionForward object..

Q 8. Write code of any Action Class?

```
package com.durgasoft;
import javax.servlet.http.*;
import org.apache.struts.action.*;
public class TestAction extends Action
{
public ActionForward execute(ActionMapping mapping, ActionForm form,
```

```
 HttpServletRequest request,HttpServletResponse response) throws Exception
  {
    return mapping.findForward("success");
  }
}
```

Q 9. What is ActionForm?

Any java class which extends from org.apache.struts.action.ActionForm is called ActionForm. An ActionForm is also called JavaBean. ActionForm maintains the session state for web application and the ActionForm object is automatically populated on the server side with data entered from a form on the client side.

Q10. What is Struts Validator Framework?

Struts Framework provides the functionality to validate the form data. It can be use to validate the data on the users browser as well as on the server side. Struts Framework emits the java scripts and it can be used validate the form data on the client browser. Server side validation of form can be accomplished by sub classing your From Bean with **DynaValidatorForm** class.
The Validator framework was developed by David Winterfeldt as third-party add-on to Struts. Now the Validator framework is a part of Jakarta Commons project and it can be used with or without Struts. The Validator framework comes integrated with the Struts Framework and can be used without doing any extra settings.

Q11. How you will display validation fail errors on jsp page?

Following tag displays all the errors:

```
<html:errors/>
```

Q12. What is RequestProcessor?

The controller is responsible for intercepting and translating user input into actions to be performed by the model. The controller is responsible for selecting the next view based on user input and the outcome of model operations. The Controller receives the request from the browser, invoke a business operation and coordinating the view to return to the client.The controller is implemented by a java servlet, this servlet is centralized point of control for the web application. In struts framework the controller responsibilities are implemented by several different components like
The ActionServlet Class
The RequestProcessor Class
The Action Class
The ActionServlet extends the **javax.servlet.http.httpServlet** class. The ActionServlet class is not abstract and therefore can be used as a concrete controller by your application.
The controller is implemented by the ActionServlet class. All incoming requests are mapped to the central controller in the deployment descriptor as follows.

```
<servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
```

```
    </servlet>
```

All request URIs with the pattern *.do are mapped to this servlet in the deployment descriptor as follows.

```
<servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
<servlet-mapping>
```

A request URI that matches this pattern will have the following form.
http://localhost:8080/mycontext/actionName.do
The preceding mapping is called extension mapping, however, you can also specify path mapping where a pattern ends with /* as shown below.

```
<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>/do/*</url-pattern>
  <url-pattern>*.do</url-pattern>
```

A request URI that matches this pattern will have the following form.
http://localhost:8080/mycontext/do/action_Name The class org.apache.struts.action.requestProcessor process the request from the controller. You can sublass the RequestProcessor with your own version and modify how the request is processed.
Once the controller receives a client request, it delegates the handling of the request to a helper class. This helper knows how to execute the business operation associated with the requested action. In the Struts framework this helper class is descended of org.apache.struts.action.Action class. It acts as a bridge between a client-side user action and business operation. The Action class decouples the client request from the business model. This decoupling allows for more than one-to-one mapping between the user request and an action. The Action class also can perform other functions such as authorization, logging before invoking business operation. the Struts Action class contains several methods, but most important method is the execute() method.
public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response) throws Exception
The execute() method is called by the controller when a request is received from a client. The controller creates an instance of the Action class if one doesn?t already exist. The strut framework will create only a single instance of each Action class in your application.
Action are mapped in the struts configuration file and this configuration is loaded into memory at startup and made available to the framework at runtime. Each Action element is represented in memory by an instance of the org.apache.struts.action. ActionMapping class. The ActionMapping object contains a path attribute that is matched against a portion of the URI of the incoming request.

```
<action>
    path= "/somerequest"
    type="com.somepackage.someAction"
    scope="request"
    name="someForm"
    validate="true"
    input="somejsp.jsp"
  <forward name="Success" path="/action/xys" redirect="true"/>
  <forward name="Failure" path="/somejsp.jsp" redirect="true"/>
</action>
```

Once this is done the controller should determine which view to return to the client. The

execute method signature in Action class has a return type org.apache. struts.action.ActionForward class. The ActionForward class represents a destination to which the controller may send control once an action has completed. Instead of specifying an actual JSP page in the code, you can declaratively associate as action forward through out the application. The action forward are specified in the configuration file.

```
<action>
      path= "/somerequest"
      type="com.somepackage.someAction"
      scope="request"
      name="someForm"
      validate="true"
      input="somejsp.jsp"
   <forward name="Success" path="/action/xys" redirect="true"/>
   <forward name="Failure" path="/somejsp.jsp" redirect="true"/>
</action>
```

The action forward mappings also can be specified in a global section, independent of any specific action mapping.

```
<global-forwards>
   <forward name="Success" path="/action/somejsp.jsp" />
   <forward name="Failure" path="/someotherjsp.jsp" />
</global-forwards>
```


Q13. How you will handle exceptions in Struts?

In Struts you can handle the exceptions in two ways:

a) Declarative Exception Handling: You can either define global exception handling tags in your struts-config.xml or define the exception handling tags within <action>..</action> tag.

Example:

```
<exception
    key="database.error.duplicate"
    path="/UserExists.jsp"
    type="mybank.account.DuplicateUserException"/>
```

b) Programmatic Exception Handling: Here you can use try{}catch{} block to handle the exception.

Q14. What are the different kinds of actions in Struts?

The different kinds of actions in Struts are:
ForwardAction, IncludeAction, DispatchAction, LookupDispatchAction, SwitchAction

Q15. What is DispatchAction?

The DispatchAction class is used to group related actions into one class. Using this class, you can have a method for each logical action compared than a single execute method. The DispatchAction dispatches to one of the logical actions represented by the methods. It picks a method to invoke based on an incoming request parameter. The value of the incoming parameter is the name of the method that the DispatchAction will invoke.

Q16. How to use DispatchAction?

To use the DispatchAction, follow these steps :

1. Create a class that extends DispatchAction (instead of Action)
2. In a new class, add a method for every function you need to perform on the service – The method has the same signature as the execute() method of an Action class.
3. Do not override execute() method – Because DispatchAction class itself provides execute() method.
4. Add an entry to struts-config.xml

Q17. What is LookupDispatchAction?

The LookupDispatchAction is a subclass of DispatchAction. It does a reverse lookup on the resource bundle to get the key and then gets the method whose name is associated with the key into the Resource Bundle.

Q18. What is the use of LookupDispatchAction?

LookupDispatchAction is useful if the method name in the Action is not driven by its name in the front end, but by the Locale independent key into the resource bundle. Since the key is always the same, the LookupDispatchAction shields your application from the side effects of I18N.

Q19. What is difference between LookupDispatchAction and DispatchAction?

The difference between LookupDispatchAction and DispatchAction is that the actual method that gets called in LookupDispatchAction is based on a lookup of a key value instead of specifying the method name directly.

Q20. What is SwitchAction?

The SwitchAction class provides a means to switch from a resource in one module to another resource in a different module. SwitchAction is useful only if you have multiple modules in your Struts application. The SwitchAction class can be used as is, without extending.

Q21. What if <action> element has <forward> declaration with same name as global forward?

In this case the global forward is not used. Instead the <action> element's <forward> takes precendence.

Q22. What is difference between ActionForm and DynaActionForm?

An ActionForm represents an HTML form that the user interacts with over one or more pages. You will provide properties to hold the state of the form with getters and setters to access them. Whereas, using DynaActionForm there is no need of providing properties to hold the state. Instead these properties and their type are declared in the struts-config.xml. The DynaActionForm bloats up the Struts config file with the xml based definition. This gets annoying as the Struts Config file grow larger.
The DynaActionForm is not strongly typed as the ActionForm. This means there is no compile time checking for the form fields. Detecting them at runtime is painful and makes you go through redeployment.
ActionForm can be cleanly organized in packages as against the flat organization in the Struts Config file.
ActionForm were designed to act as a Firewall between HTTP and the Action classes, i.e. isolate and encapsulate the HTTP request parameters from direct use in Actions. With DynaActionForm, the property access is no different than using request.get Parameter(..).

- DynaActionForm construction at runtime requires a lot of Java Reflection (Introspection) machinery that can be avoided.

Q23. What is the life cycle of ActionForm?

The lifecycle of ActionForm invoked by the RequestProcessor is as follows:

- Retrieve or Create Form Bean associated with Action
- "Store" FormBean in appropriate scope (request or session)
- Reset the properties of the FormBean
- Populate the properties of the FormBean

- Validate the properties of the FormBean
- Pass FormBean to Action

Q24.What are the important tags of struts-config.xml ?

```
<struts-config>
  <!-- ========== Form Bean Definitions ============ -->
<form-beans>
<form-bean name="login" type=" LoginForm" />
  </form-beans>
  <!-- ========== Global Forward Definitions ========= -->
  <global-forwards>
  </global-forwards>
  <!-- ========== Action Mapping Definitions ======== -->
  <action-mappings>
    <action
        path="/login"
        type="LoginAction" >
        </action>
  </action-mappings>
<!-- ========== Properties Definitions ============ -->
<message-resources parameter="MessageResources" />
<!-- ========== Validator framework Definitions ============ -->
<plug-in className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property
           property="pathnames"
            value="/org/apache/struts/validator/validator-rules.xml,
        /WEB-INF/validation.xml"/>
   </plug-in>
</struts-config>
```

Q25. What are the core classes of the Struts Framework?

A: Core classes of Struts Framework are ActionForm, Action, ActionMapping, Action Forward, ActionServlet etc.

Q26. What is action mappings?

An action mapping is a configuration file entry that, in general, associates an action name with an action. An action mapping can contain a reference to a form bean that the action can use, and can additionally define a list of local forwards that is visible only to this action.

Q27. Describe validate() and reset() methods ?

validate () and reset() methods defined inActionForm class.
validate() : Used to validate properties after they have been populated; Called before FormBean is handed to Action. Returns a collection of ActionMessage as ActionErrors. Following is the method signature for the validate() method.
public ActionErrors validate(ActionMapping mapping, HttpServletRequest request)
reset(): reset() method is called by Struts Framework with each request that uses the defined ActionForm. The purpose of this method is to reset all of the ActionForm's data members prior to the new request values being set.
public void reset() {}

Q28. Give the Details of XML files used in Validator Framework?
The Validator Framework uses two XML configuration files **validator-rules.xml** and **validation.xml**. The **validator-rules.xml** defines the standard validation routines, these are reusable and used in **validation.xml**. to define the form specific validations. The **validation.xml** defines the validations applied to a form bean.

Q29. How you will enable front-end validation based on the xml in validation.xml?
The <html:javascript> tag to allow front-end validation based on the xml in validation.xml. For example the code: <html:javascript formName="logonForm" dynamicJavascript="true" staticJavascript="true" /> generates the client side java script for the form "logonForm" as defined in the validation.xml file. The <html:javascript> when added in the jsp file generates the client site validation script.

Q30. What is the difference between perform() and execute() methods?
perform() method defined in Struts 1.0. but it is was deprecated in the Struts Version 1.1. In Struts 1.x, Action.perform() is the method called by the ActionServlet. This is typically where your business logic resides, or at least the flow control to your JavaBeans and EJBs that handle your business logic. As we already mentioned, to support declarative exception handling, the method signature changed in perform. Now execute just throws Exception. Action.perform() is now deprecated; however, the Struts v1.1 ActionServlet is smart enough to know whether or not it should call perform or execute in the Action, depending on which one is available.

Q31. What are the various Struts tag libraries?
Struts is very rich framework and it provides very good and user friendly way to develop web application forms. Struts provide many tag libraries to ease the development of web applications. These tag libraries are:
*** Bean tag library** - Tags for accessing JavaBeans and their properties.
*** HTML tag library** - Tags to output standard HTML, including forms, text boxes, checkboxes, radio buttons etc..
*** Logic tag library** - Tags for generating conditional output, iteration capabilities and flow management
*** Tiles or Template tag library** - For the application using tiles
*** Nested tag library** - For using the nested beans in the application

Q32. What are the difference between <bean:message> and <bean:write>?
<bean:message>: This tag is used to output locale-specific text (from the properties files)

from a MessageResources bundle.
<bean:write>: This tag is used to output property values from a bean. <bean:write> is a commonly used tag which enables the programmers to easily present the data.

Q33. What are difference between ActionErrors and ActionMessage?

ActionMessage: A class that encapsulates messages. Messages can be either global or they are specific to a particular bean property.
Each individual message is described by an ActionMessage object, which contains a message key (to be looked up in an appropriate message resources database), and up to four placeholder arguments used for parametric substitution in the resulting message.

ActionErrors: A class that encapsulates the error messages being reported by the validate() method of an ActionForm. Validation errors are either global to the entire ActionForm bean they are associated with, or they are specific to a particular bean property (and, therefore, a particular input field on the corresponding form).

Q34. What is the use of ForwardAction?

The ForwardAction class is useful when you're trying to integrate Struts into an existing application that uses Servlets to perform business logic functions. You can use this class to take advantage of the Struts controller and its functionality, without having to rewrite the existing Servlets. Use ForwardAction to forward a request to another resource in your application, such as a Servlet that already does business logic processing or even another JSP page. By using this predefined action, you don't have to write your own Action class. You just have to set up the struts-config file properly to use ForwardAction.

Q35. What is IncludeAction?

The IncludeAction class is useful when you want to integrate Struts into an application that

uses Servlets. Use the IncludeAction class to include another resource in the response to the request being processed.

Q36. What are the steps need to use DynaActionForm?

Using a DynaActionForm instead of a custom subclass of ActionForm is relatively straightforward. You need to make changes in two places:
In struts-config.xml: change your <form-bean> to be an org.apache.struts.action.Dyna ActionForm instead of some subclass of ActionForm

- <form-bean name="loginForm"
- type="org.apache.struts.action.DynaActionForm" >
  <form-property name="userName" type="java.lang.String"/>
  <form-property name="password" type="java.lang.String" />
  </form-bean>
- In your Action subclass that uses your form bean:
  - import org.apache.struts.action.DynaActionForm
  - downcast the ActionForm parameter in execute() to a DynaActionForm
  - access the form fields with get(field) rather than getField()

Q.37 In struts what happens if made any changes in actionservlet?

The ActionServlet plays the role of controller wich is responsible for handling the request and selecting the correct Application Module and storing ApplicationConfig and MessageResource bundle in the request object.
If we modify the ActionServlet the Controller may or may not work what happens that depends on your modification, You have not specify whether you want to create your own custom ActionServlet by extending ActionServlet and overriding the methods in it or what exactly you want to modify.