Developing Applications Using Java Web Frameworks

Session - 1

Introduction to Struts Framework

Objectives

- Explain Model-View-Controller (MVC) architecture
- Explain the role of components involved in MVC architecture
- Explain JSP Model 1 and JSP Model 2 approaches
- Describe Struts framework
- List the features and components of Struts 1 framework
- Explain the architecture of Struts 1 framework
- Explain the process of developing Web applications using Struts 1 components

Introduction 1-2

- Powerful Websites have become an absolute necessity to everyone, be it an individual or a big corporation.
  - Example: A **Bank Website** containing interactivity and dynamic options based on customers expectations.

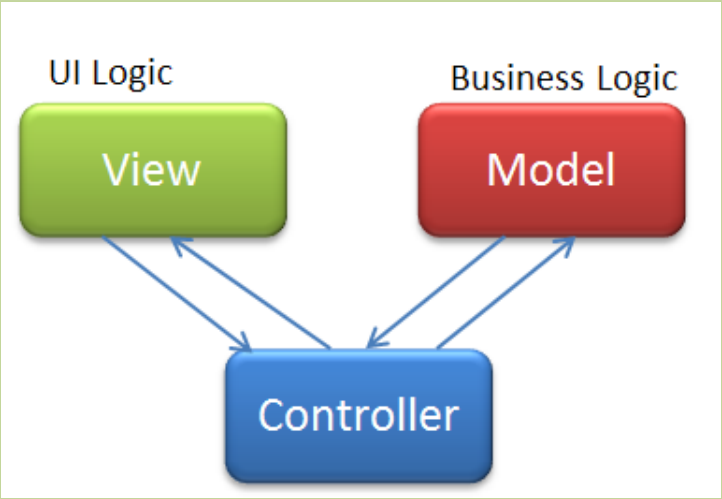
- How do architects involved in Web application design complex such Websites?
  - By adopting **Design Patterns**.

- A **Design Pattern** is the one which describes a proven solution to a recurring problem in application development.

Introduction 2-2

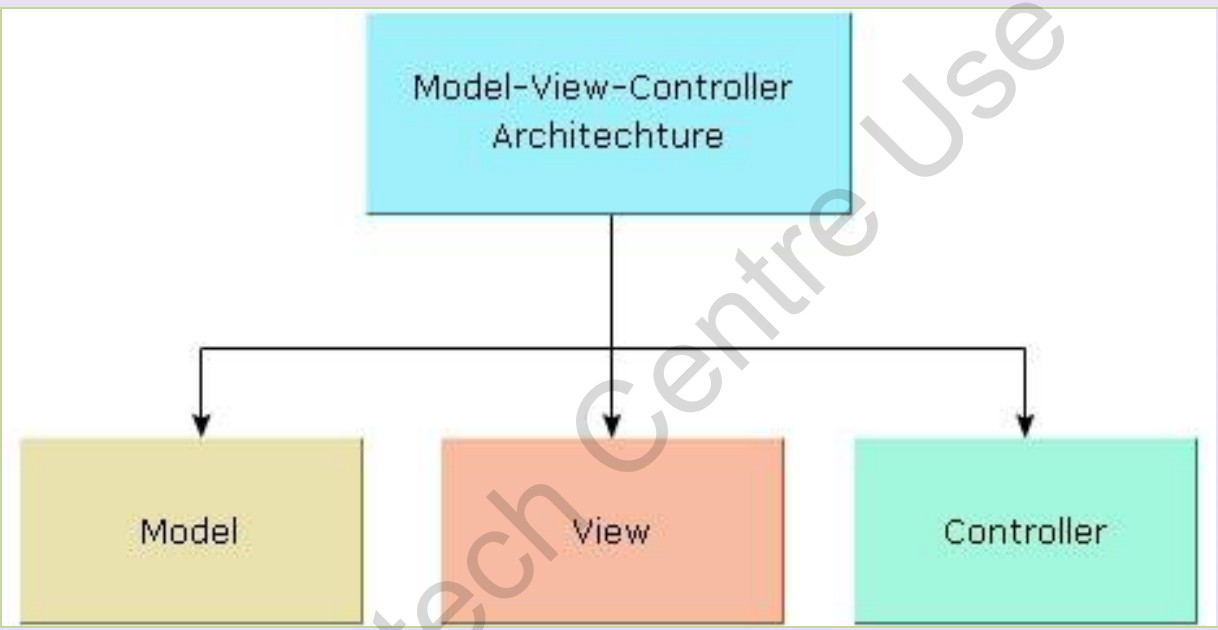
- The most widely acceptable Website designing pattern is the **Model-View-Controller (MVC)**.

- **Model-View-Controller (MVC)**
  - Is an architectural pattern whose main concern is to separate data from the user interface.
  - Changes to the user interface can be made without affecting the underlying data handling logic.



MVC Architectural Pattern 1-2

- Following figure shows the MVC architectural pattern.



- **Controller** - Defines how the user interfaces react to the user input.

MVC Architectural Pattern 2-2

- The relationship between the three components of MVC:

View and Controller

- Creation and selection of views is the responsibility of the controller.

Model and View

- The view is dependent on the model.
- If any change is to be made to the model, then there may be a requirement to make parallel changes in the view also.

Model and Controller

- The controller is dependent on the model.
- If any change is to be made to the model, then there may be a requirement to make parallel changes in the controller also.

Role of Components in MVC

- The role of the three components of MVC architecture in developing Web applications:

Model

- Encapsulates the domain logic of an application.
- Manages information and notifies observers whenever that information changes.
- Contains only data and functionality that are related by a common purpose.
- Makes it very easy to map real-world entities into a model.

View

- Obtains data from the model and presents it to the user.
- Represents the output and/or input of the application.
- Has free access to the model, but should not change the state of the model.
- It reads data from the model using query methods provided by the model.

Controller

- Receives and translates input to requests on the Model or View.
- Provides the means by which a user can interact with the application.
- Accepts input from the user and instructs the Model and View to perform actions based on that input.
- Responsible for calling methods on the model that changes the state of the model.

Java Web Development Models

- With the introduction of JSP technology, Sun Microsystems provided a road map for developing JSP-based Web applications.
- Java platform has defined two models which provided two approaches for designing Java-based Web applications.
- These are as follows:

Model 1

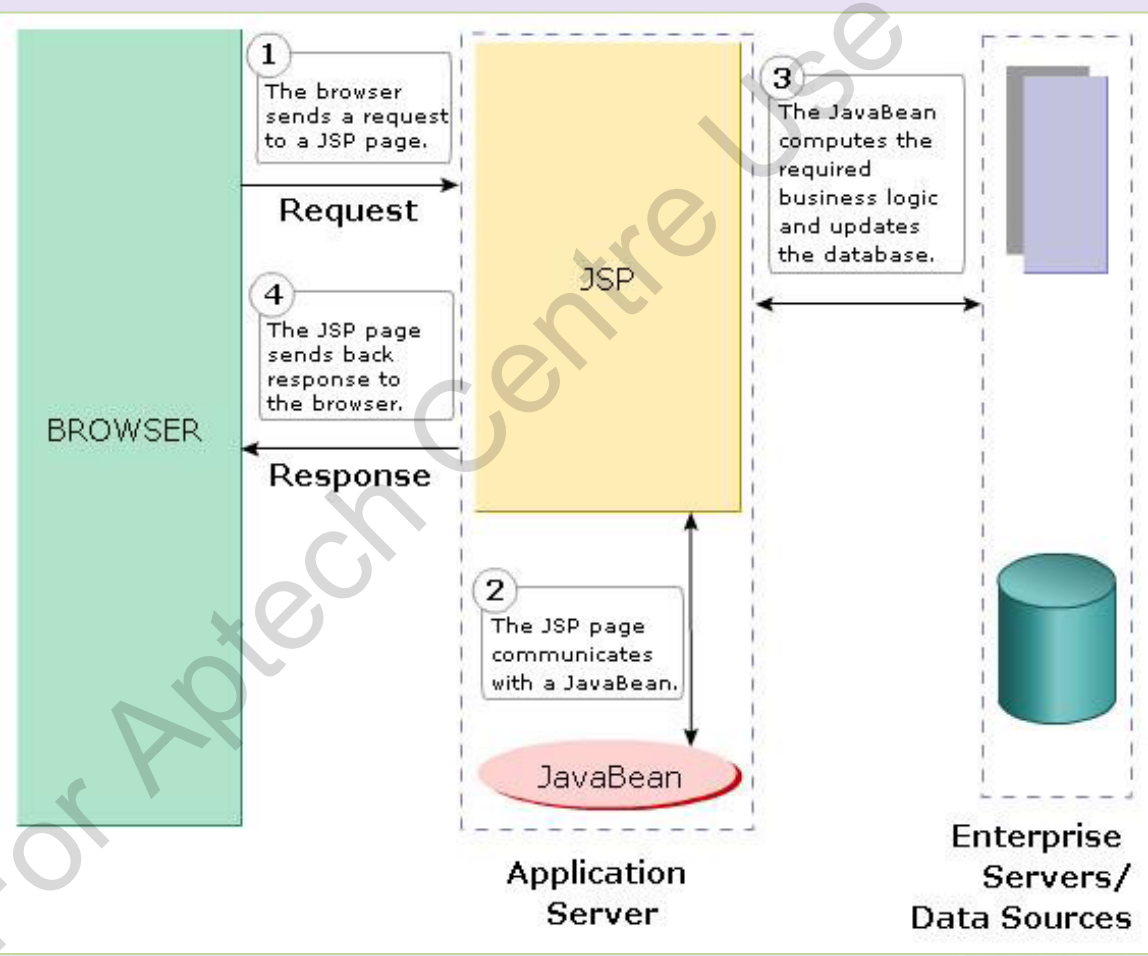
Model 2

JSP Model 1 Architecture 1-3

- Is a page-centric architecture.
- Provides JSP page as the main component.
  - JSP page is responsible for processing requests and sending back replies to the clients.
  - JSP page is also responsible for extracting the HTTP request parameters, invoking the business logic, and handling the HTTP session.

JSP Model 1 Architecture 2-3

- Following figure shows JSP Model 1 architecture.



JSP Model 1 Architecture 3-3

- The disadvantages of JSP Model 1 Architecture are as follows:

It suffers from navigation problem

- If you change the name of a JSP page that is referenced by other pages, you must change the name in many locations.

It is difficult to maintain an application and is inflexible

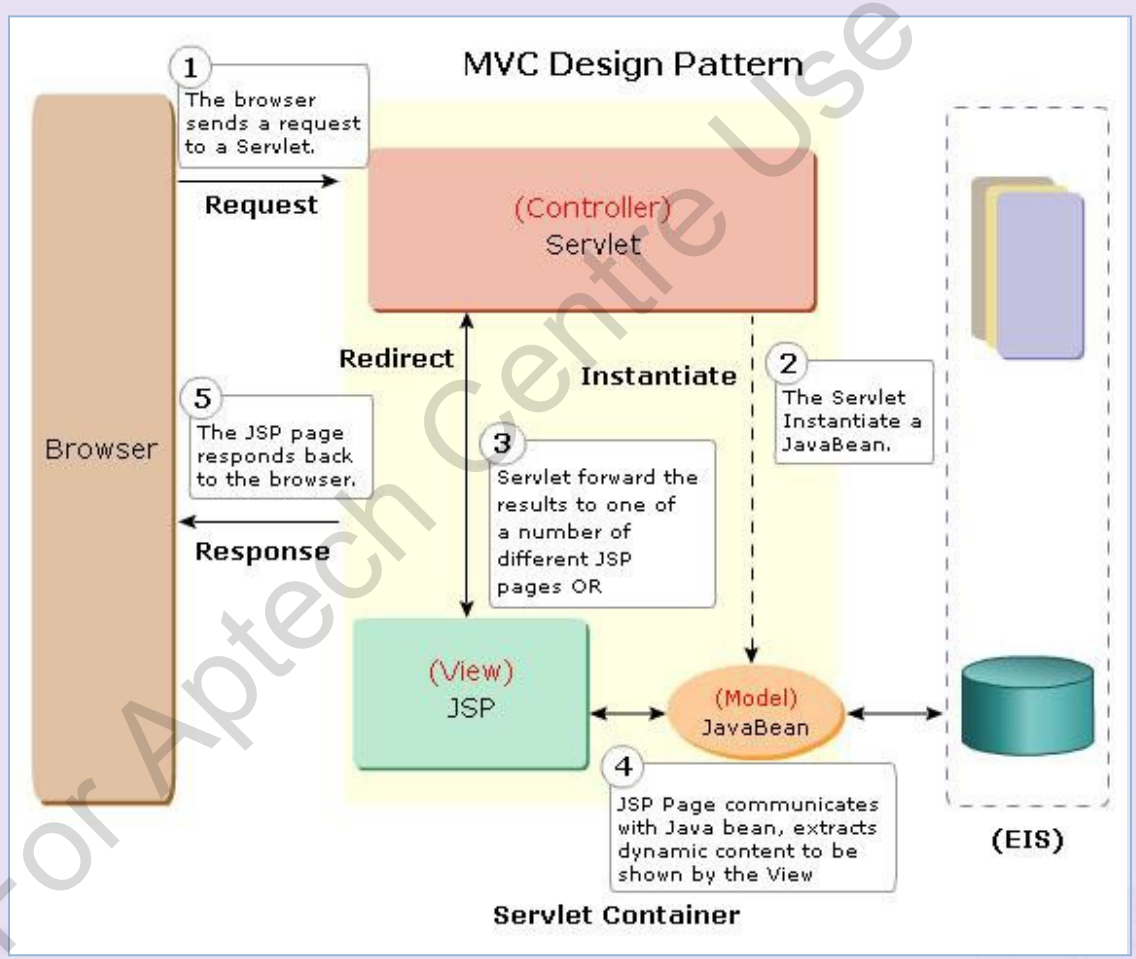
- If an application requires a number of request processing events then, it will lead to a large Java code being embedded within the JSP page.

JSP Model 2 Architecture 1-3

- Is a Servlet-centric architecture.
- Separates the 'Business Logic' from the 'Presentation Logic'.
- Provides **Controller Servlet** as main component.
  - Responsibility of the Controller Servlet is to process the incoming request and instantiate a Model - a Java object or a bean to compute the business logic.
  - Responsible for deciding to which JSP page the request should be forwarded.

JSP Model 2 Architecture 2-3

- Following figure shows the Model 2 architecture:



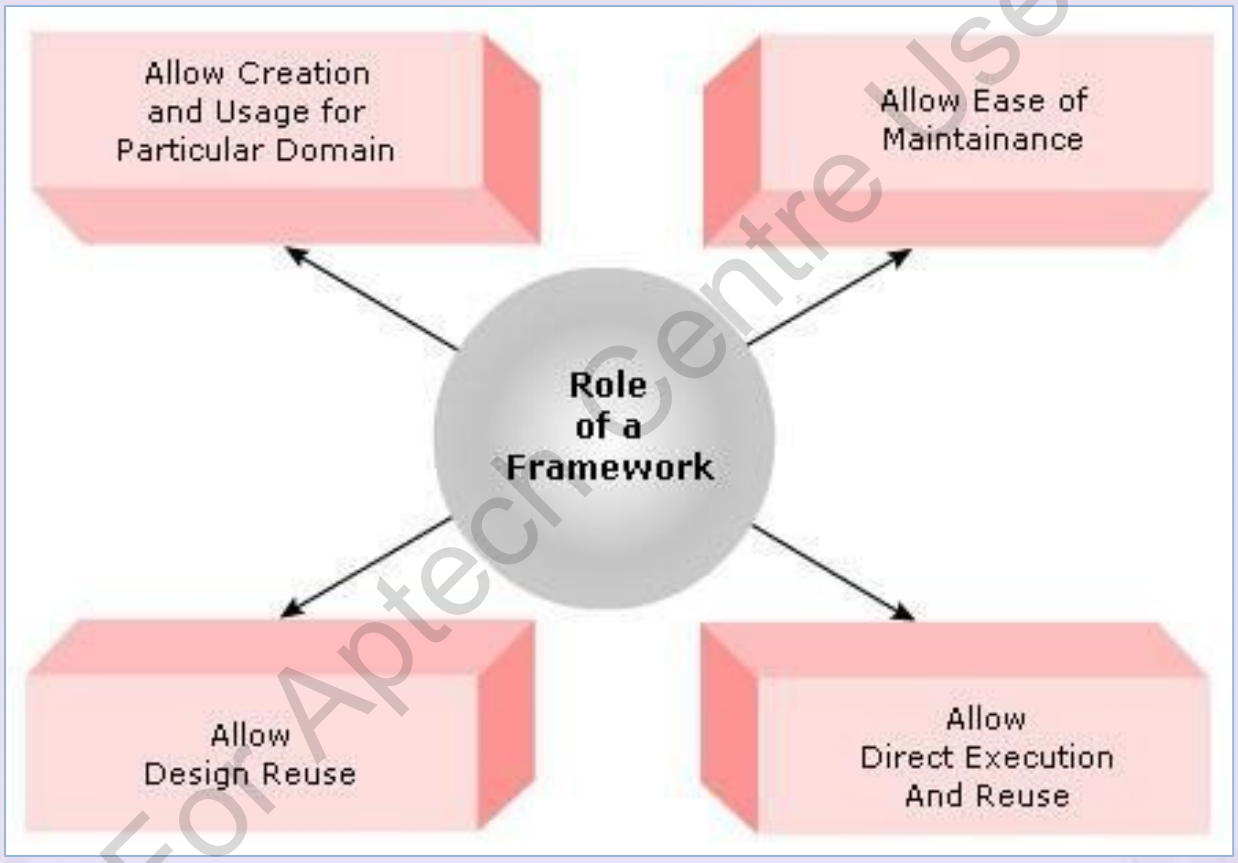
JSP Model 2 Architecture 3-3

❑ Advantages and disadvantages of JSP Model 2 Architecture:

Advantages	Disadvantages
• Easier to build, maintain, and extend. • Provides a single point of control for security and logging. • Encapsulates incoming data into a form usable by the back-end MVC model.	• Increases the design complexity as it introduces some extra classes/code due to the separation of model, view, and controller components.

Role of a Framework

- Following figure shows role of a framework:



Characteristics of a Framework

- A framework consists of various classes and components.
- Framework classes or component gives an abstraction of a specific concept.
- Framework gives a definition of how these classes or components work with each other to provide a solution to a problem.
- Framework components and classes can be reused.
- Framework provides an organized pattern of an application.

Description of Controller

- Frameworks are needed to incorporate the following features in software design process:

Modularity

- It is achieved by encapsulating the application specific detail.

Extensibility

- An ease with which a new behavior can be added depending upon the application domain.

Reusability

- It is enhanced by frameworks by defining generic components that can be reapplied.

Inversion of control

- It reduces the tight coupling between associating objects.
- Create separate event handler objects.

Introduction to Struts 1-2

- Is an open source Java-based framework provided by Apache Software Foundation.
- Simplifies the development of Java Enterprise Edition (Java EE) Web applications based on MVC architecture.
- Extends the Java Servlet API.
- Has become the default standard for building Web applications in Java.
- Resides in the Web tier.
- Managed by the Web or Servlet container residing on the Web server.

Introduction to Struts 2-2

- Struts framework is available in two versions.
  - Struts 1 is the first version of Apache Struts framework.
  - In the year 2007, Apache released Struts 2 framework combining the features of WebWork framework to offer new enhancements and refinements in the original Struts framework.

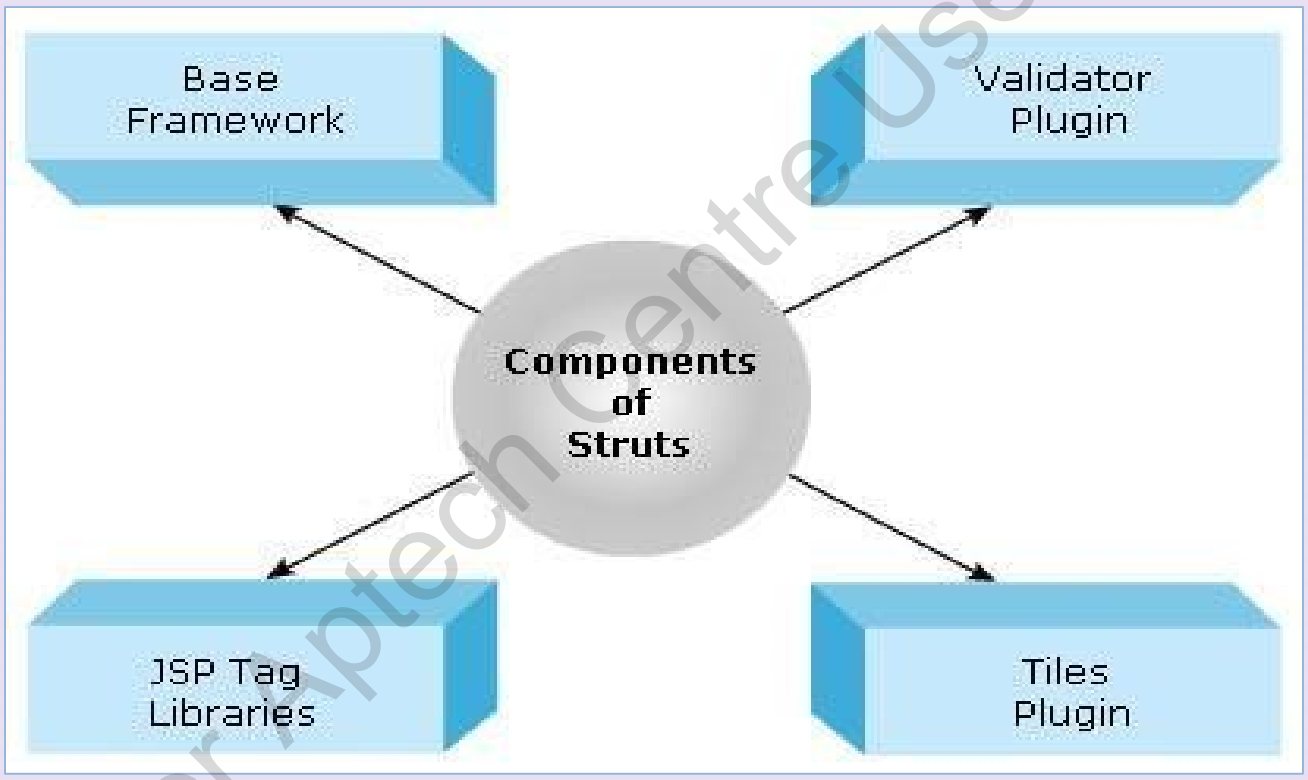
Features of Struts

- The main features of Struts are:

Code Extensibility	• Struts values are represented in XML or in property files. • Changes can be made to the file without recompiling the Java code.
Support for localization and internationalization	• Struts support localization and internationalization in the form of ResourceBundle.
Simpler request process mechanism	• Struts provides several utilities, such as `StrutValidatorUtil`, `MessageResource` class, and so on.
Model-view communication	• Struts provide a set of custom JSP tags which allows you to create HTML forms. • These forms directly associates with Java Bean component.
Input validation	• Struts provides built-in capability for validation of form values.

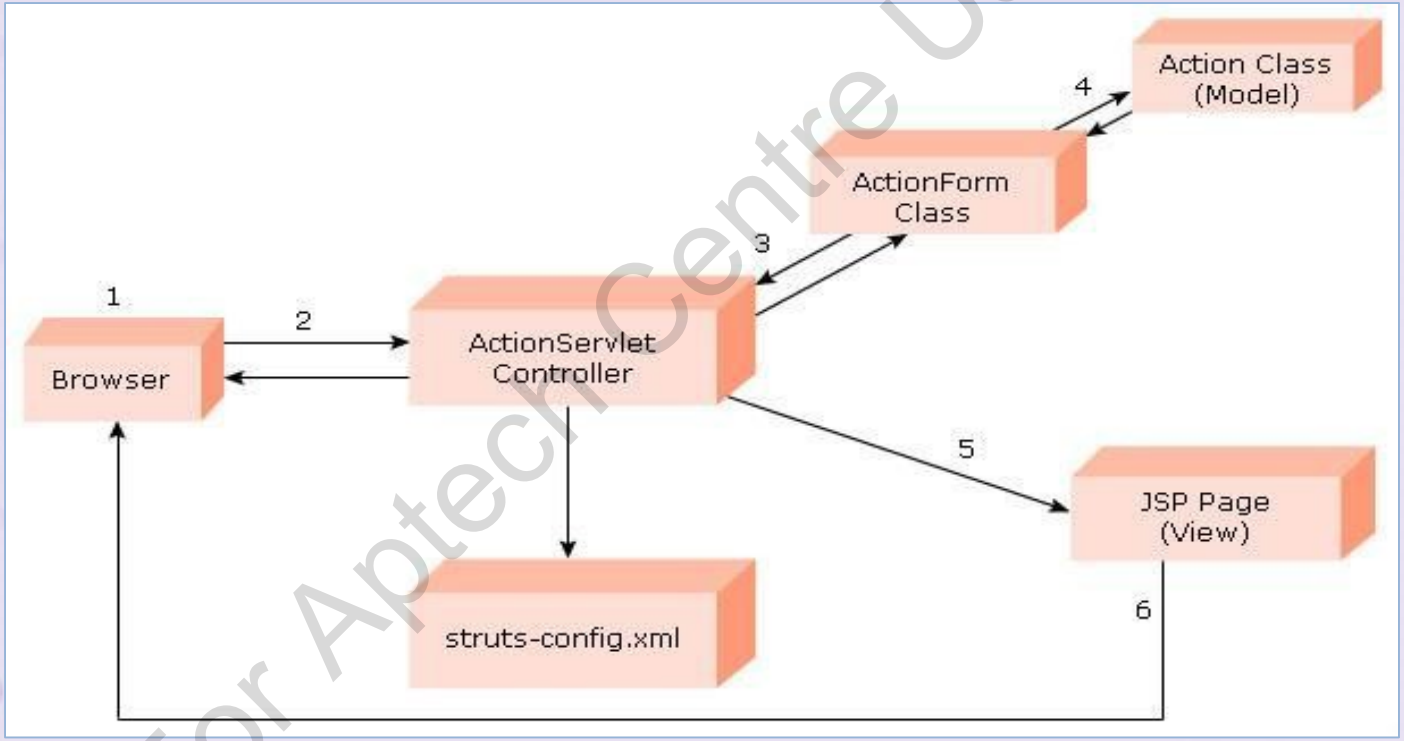
Components of Struts Framework

- Following figure shows the major constituent of Struts framework:



Struts 1 Architecture

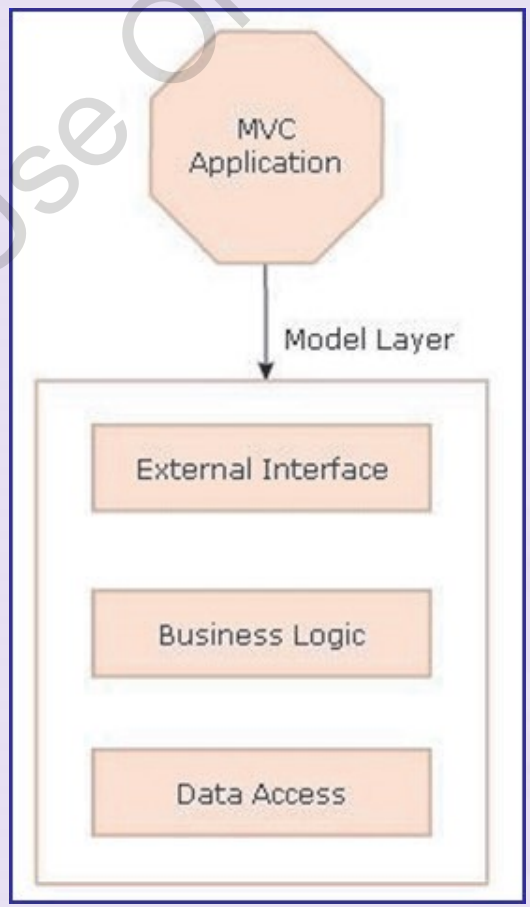
- Struts framework defines all the three MVC components.
- Following figure shows Struts MVC components:



Struts Model Component

- Struts architecture does not provide any specific constraints for the Model component.
- The model layer may be broken down into three sublayers.

Figure alongside depicts the Struts model components.



Struts View Component

- Components of a Struts View are as follows:

JSP Pages

- Provide JSP tag library which allow creating HTML forms for capturing data and also displaying the data.
- Contain static HTML and JSP library tags to generate dynamic HTML page.

Form Beans

- Provide a channel for data transfer between the view and control layers of Struts application.

JSP Tag Libraries

- Provide a means to create HTML forms whose data will be captured and stored in Form beans.
- Contain utility tags which help to implement conditional logic, iteration, and many more.

Resource bundles

- Allow internationalization of Java application.

Struts Controller Component 1-2

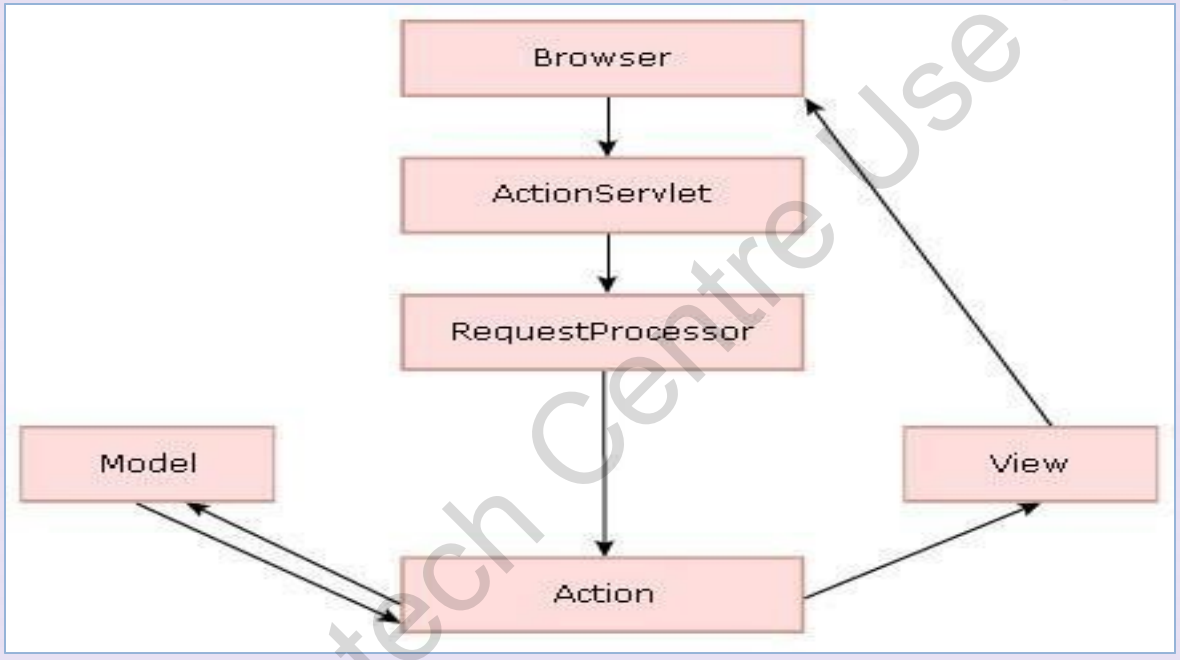
- The core of Struts framework is Struts Controller Servlet called **ActionServlet**.

- **ActionServlet class:**
  - Handles run-time events in accordance with a set of rules.
  - Delegates its processing of a request to a RequestProcessor class.

- **RequestProcessor class:**
  - Selects the Form Bean, populates the Form Bean, performs validation.
  - The Struts framework then access with the Action class.

Struts Controller Component 2-2

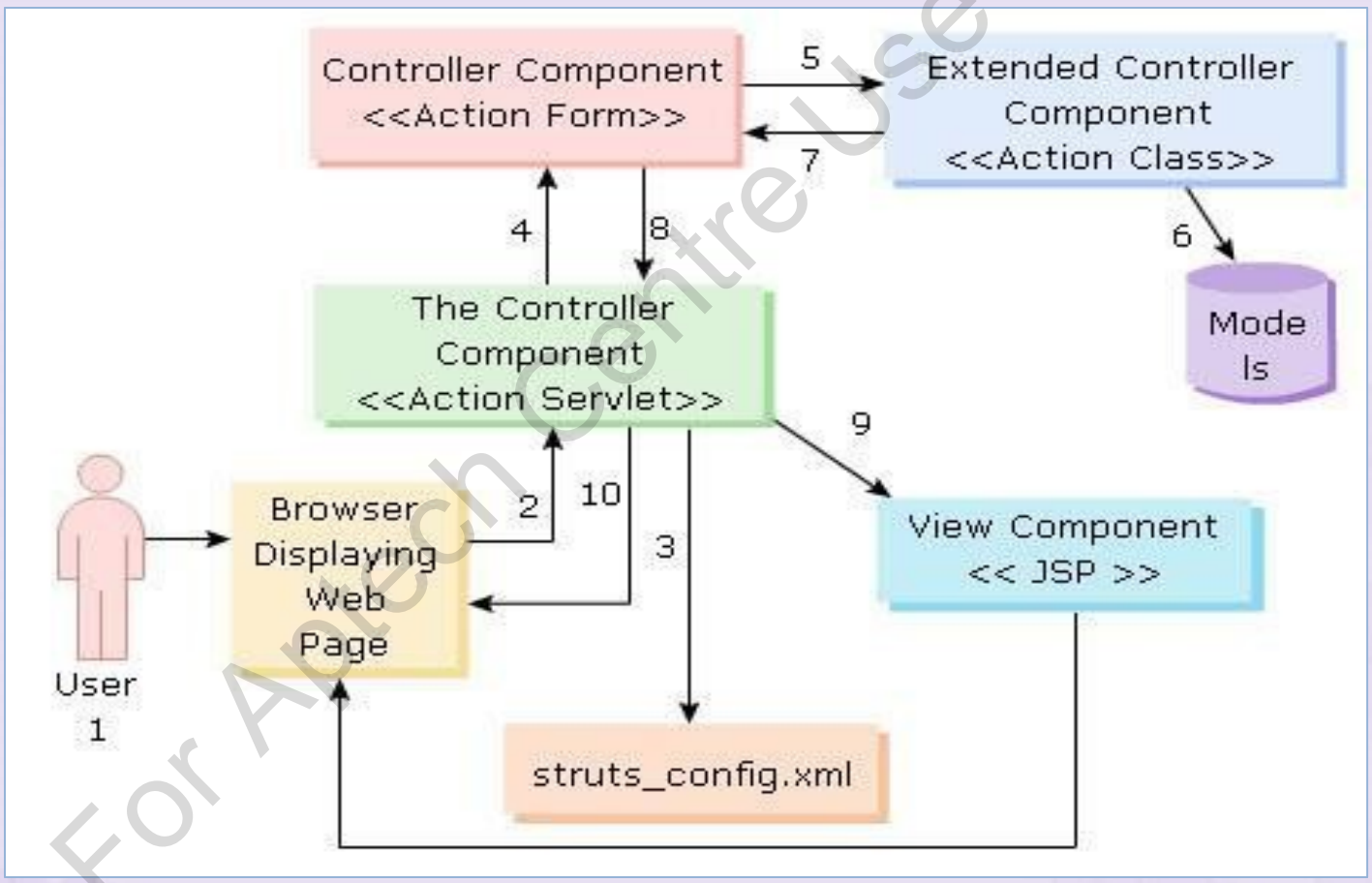
- Following figure shows the Struts Controller components:



- Struts follow same control flow as that of the MVC architecture.

Struts 1 Application Control Flow

- Following figure shows the Struts control flow:



struts-config.xml

- The `struts-config.xml` configuration file is a link between the View and Model components in the Web:

struts-config	• This is the root node of the configuration file.
Form-beans	• This section allows you to map your `ActionForm` subclass to a name. The form name is used as alias for the other elements in the `struts-config.xml` file.
global forwards	• This section maps a page on your Web application to a name. You can use this name to refer to the actual page.
action-mappings	• This is where you declare form handlers and they are also known as action mappings.
controller	• This section configures Struts Controller Servlet.
plug-in	• This section tells Struts where to find your properties files. • It contains prompts and error messages.

Developing Struts 1 Web Application 1-2

- **Step 1: Configure Controller – ActionServlet**
  - `org.apache.struts.action.ActionServlet` class is the core class of all Struts based application.
  - The purpose of the `ActionServlet` class is to receive all incoming HTTP requests for the application and determine which `Action` will process the incoming request.
- The developer needs to configure the `ActionServlet` class in `web.xml` file of the Web application.

The `web.xml` file is an deployment descriptor XML file with several elements. It is this file in which the `ActionServlet` class is configured.

Developing Struts 1 Web Application 2-2

- Following code snippet shows how to configure the Servlet name and class in the `web.xml` file.

```
<web-app>
<! -- ActionServlet Configuration -->
<Servlet>
  <Servlet-name>actionExample</Servlet-name>
  <Servlet-
class>org.apache.struts.action.ActionServlet</Servlet-class>
</Servlet>
<Servlet-mapping>
  <Servlet-name>actionExample</Servlet-name>
  <url-pattern>*.do</url-pattern>
</Servlet-mapping>
</web-app>
```

- The pattern matches the ActionServlet named '`actionExample`' and services all requests having an extension of `.do`.

Developing Action 1-9

Step 2: Developing an Action class

- `org.apache.struts.action` package provides the `Action` class.
- `Action` class does not contain business logic, rather it is designed to delegate business logic to the Model component.
- `Action` class is acting as a bridge between the View and Model layer.
- When a request is made to the `ActionServlet` class with a given path, the appropriate action for processing the request is invoked.

Developing Action 2-9

- Following figure shows the class diagram of `Action` class.

Developing Action 3-9

- To use the `Action` class, developer has to subclass and overwrite the `execute()` method.
- The `execute()` method has two functions:

execute() method	
It performs the business logic for the application.	It helps Framework to determine where it should next route the request.

- When a user performs an action, the `execute()` method, Action class is invoked to process the request based on user's action.
- The `execute()` method may throw any business logic exceptions.

Developing Action 4-9

- **Syntax:**

```
public ActionForward execute(ActionMapping map,
ActionForm form, javax.servlet.ServletRequest req,
javax.servlet.ServletResponse resp) throws
java.lang.Exception
```

where,

- `map`: represents an object of ActionMapping
- `form`: refers to an object of `ActionForm` class
- `req`: refers to the HTTP request object
- `resp`: refers to the HTTP response object
- `Exception`: raises an exception only when the business code throws an exception

Developing Action 5-9

- The Struts Framework defines two implementations for the `execute()` method:

Implementation - I	Implementation - II
Define custom Actions that are non-HTTP specific.	Define custom Actions that are HTTP specific.
It is to be overridden in order to service request that are not HTTP specific.	It is to be overridden in order to service HTTP specific requests.

Developing Action 6-9

- Following table shows the parameters of `Action.execute()` method:

Parameter	Description
ActionMapping	It contains all of the deployment information for a particular Action bean.
ActionForm	It represents the Form inputs containing the request parameters from the View referencing this Action bean.
ServletRequest or HttpSevletRequest	It is a reference to current request object either Http specific or non-Http specific respectively.
ServletResponse or HttpServletResponse	It is a reference to current response object either Http specific or non-Http specific respectively.

Developing Action 7-9

- To derive an Action class, following steps are to be executed:

Create a class that extends the Action class of `org.apache.struts.action` package.

Override the `execute()` method in order to specify the desired business logic.

It returns the `ActionForward` object which is used to route the request to the specified view.

To describe the new action add an `<action>` element in the Struts configuration file.

Developing Action 8-9

- Following code snippet demonstrates the development of **LoginAction** class:

```
public class LoginAction extends
org.apache.struts.action.Action {
  /* forward name="success" path="" */
  private final static String SUCCESS = "success";
  private final static String FAILURE = "failure";
  /**
   * This is the action called from the Struts framework.
   * @param mapping The ActionMapping used to select this
    instance.
   * @param request The HTTP Request we are processing.
   * @param response The HTTP Response we are processing.
   * @throws java.lang.Exception
   * @return   */
```

Developing Action 9-9

```
public ActionForward execute(ActionMapping mapping, ActionForm
form,
      HttpServletRequest request, HttpServletResponse response)
throws Exception {
     LoginForm loginForm = (LoginForm) form;

   if (loginForm.getUserName().equals(loginForm.getPassword())) {
      return mapping.findForward(SUCCESS);
    } else {
      return mapping.findForward(FAILURE);
    }
  }
}
```

ActionForward Class 1-3

- It is a destination where the controller (RequestProcessor) can process the forward request.
- The `execute()` method when executed returns an object of `ActionForward` class.
- This object has been mapped to the name of the forward.
- One change to forward definition is sufficient to reflect all places in application.

ActionForward Class 2-3

- Following table shows the properties supported by an `ActionForward` class:

Property	Description
`contextRelative`	It specifies if URL value for path represents the absolute path or path is relative to the module under consideration.
`name`	It is the logical name by which the instance may be looked up in relationship to a particular ActionMapping.
`path`	It has the value that interpreted as Module-relative or context-relative URL to which the control should be forwarded or an absolute or relative URL to which control should be redirected.
`redirect`	It is set to true if the controller servlet should call `HttpServletResponse.sendRedirect()` on the associated path; otherwise should be set to false.

ActionForward Class 3-3

- Following code snippet declares the configuration of action along with the forward:

```
<action-mappings>
 <action input="/login.jsp" name="LoginForm" path="/Login"
scope="session“ type="com.example.LoginAction">
  <forward name="success" path="/success.jsp" />
  <forward name="failure" path="/failure.jsp" />
 </action>
</action-mappings>
</struts-config>
```

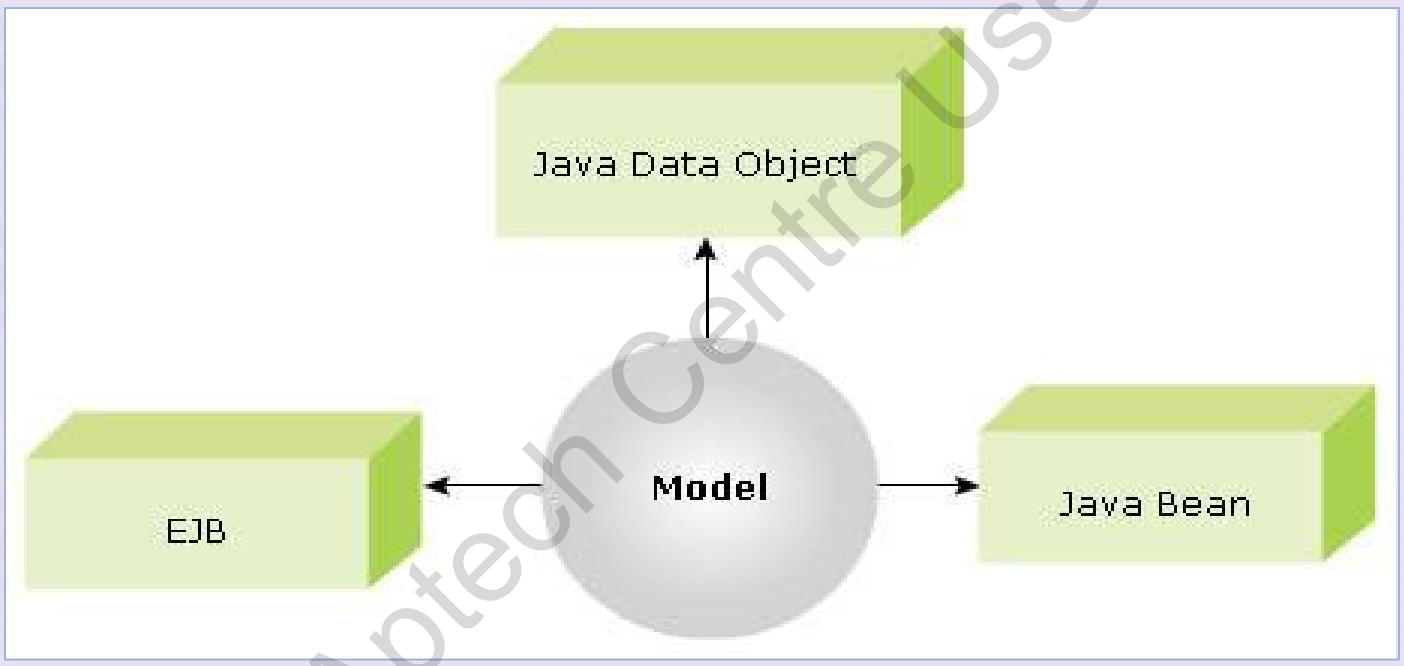
- Code Snippet defines a URL to be used as forward with local scope. Here, the action will be forwarded to the `success.jsp` page, whenever the forward named ‘`Success`’ is found.

Developing Model 1-2

- Model component of struts framework represents application's business data and the rules.
- The purpose of having a separate model component is to ensure that the model remains independent and reusable.
- The model components are accessed from the subclass of struts Action class.
- The Action sub class via its Action interface uses its Data Transfer Object to pass and retrieve data from model.
- Business specific model code should not refer to Struts code or object.

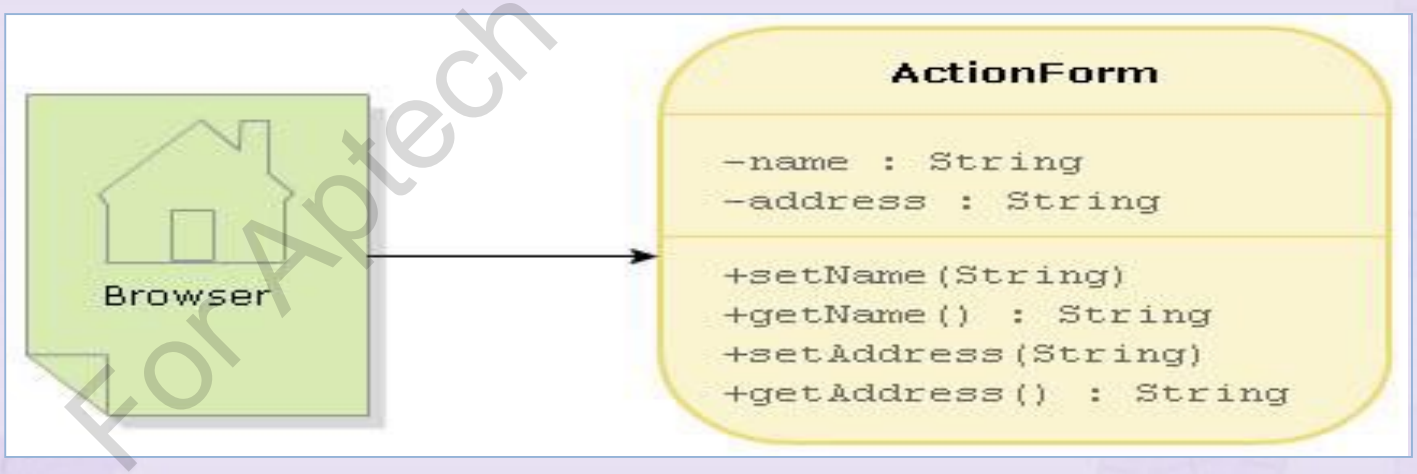
Developing Model 2-2

- Following figure shows the Struts model:



Developing ActionForm 1-6

- Struts framework supports the functionality for retrieving the data entered by Web application user.
- Store data temporarily for validation and displaying an error message for invalid data.
- JSP page provide the input fields for an HTML form.
- HTML form takes data from the `ActionForm` class and not from Java Beans.
- Following figure shows the class diagram of `ActionForm` class:



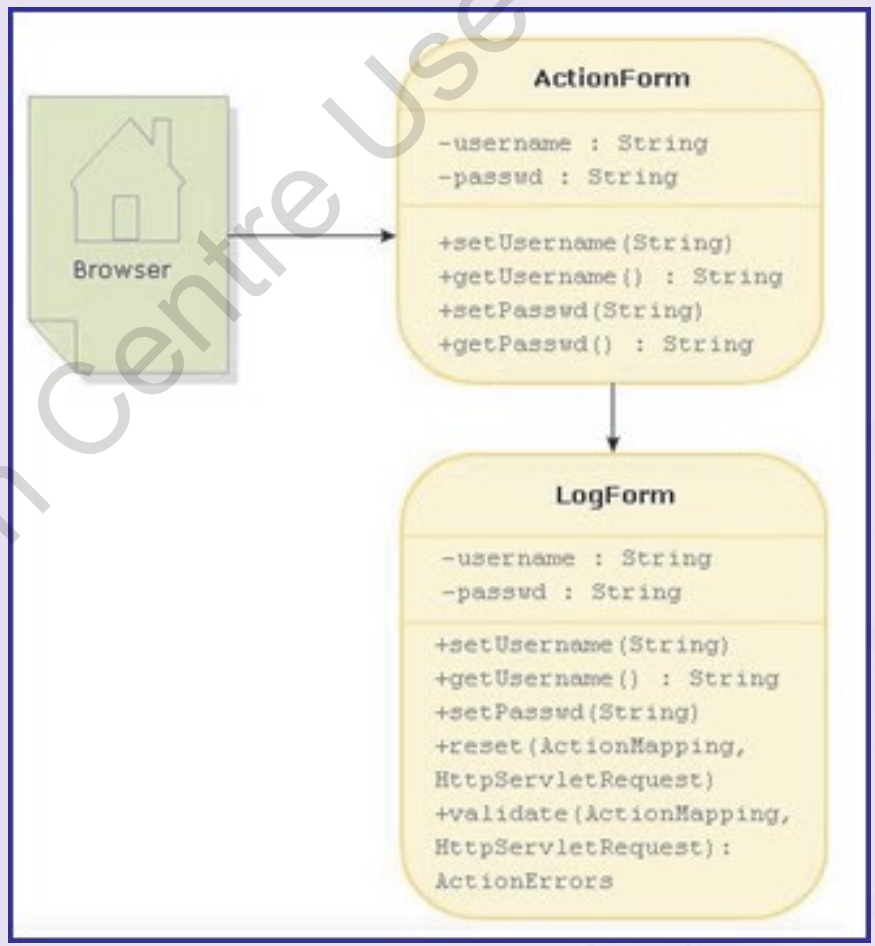
Developing ActionForm 2-6

- The `org.apache.struts.action.ActionForm` is the abstract base class supporting this functionality.
- Basic Java beans with 'get' and 'set' methods are defined for each of their properties and are used for supporting the functionality.
- The abstract `ActionForm` class has two methods which a subclass may override for customized `ActionForm` class:

validate	reset
• Is called by the RequestProcessor class. • Performs data validation on ActionForm attributes before data is sent to Action class.	• Is used to reset the ActionForm attributes. • Is called for each new request, before the `ActionForm` is populated.

Developing ActionForm 3-6

- Following figure shows the `LoginForm` class extending from `ActionForm`:



Developing ActionForm 4-6

- Following code snippet demonstrates the development of `LoginForm` class:

```
public class LoginForm extends org.apache.struts.action.ActionForm {

    private String userName;
    private String password;

    public LoginForm() {
        super();
        // TODO Auto-generated constructor stub
    }

/**
* This is the action called from the Struts framework.
    * @param mapping The ActionMapping used to select this instance.
    * @param request The HTTP Request we are processing.
    * @return
    */
```

- The `LoginForm` class declares two properties - the `userName` and `password` corresponding to the form fields.

Developing ActionForm 5-6

```
public ActionErrors validate(ActionMapping mapping, HttpServletRequest request) {
    ActionErrors errors = new ActionErrors();
    if (userName == null || userName.length() < 1) {
      errors.add("userName", new ActionMessage("error.userName.required"));
      // TODO: add 'error.name.required' key to your resources
    }
if (password == null || password.length() < 1) {
errors.add("password", new ActionMessage("error.password.required"));
// TODO: add 'error.name.required' key to your resources
}
return errors;
 }
 /**
* @return the userName
 */
  public String getUserName() {
    System.out.println("Inside getter "+userName);
    return userName;
  }
```

- The `validate()` method checks if the `userName` and `password` have been assigned value or not.
- It returns an error if these attributes have null value.

Developing ActionForm 6-6

```
 /**
   * @param userName the userName to set  */
public void setUserName(String userName) {
    System.out.println("Inside setter "+userName);
    this.userName = userName;
}
   /**
   * @return the password  */
  public String getPassword() {
    return password;
  }
   /**
   * @param password the password to set */
  public void setPassword(String password) {
    this.password = password;
  }
}
```

- Getter and setter method is specified for `userName` and `password` attributes.

JSP Tag Libraries 1-4

- The JSP tag libraries are fundamental building blocks in Struts applications:
  - Since, they provide a convenient mechanism for creating HTML forms whose data will be captured in Form Beans and for displaying data stored in Form Beans.
- The list of Struts tag libraries are as follows:
  - **HTML**
  - **Bean**
  - **Logic**

JSP Tag Libraries 2-4

- Purpose of HTML and Nested tags is as follows:

HTML	Nested
• It is used to generate HTML forms that interact with the Struts APIs. • The tags in this library can automatically populate form controls with data from Form Beans. • Save time and several lines of code.	• It is used to allow arbitrary levels of nesting of HTML, Bean, and Logic tags.

JSP Tag Libraries 3-4

- Purpose of Bean and Logic tags are as follows:

Bean	Logic
• It is used to work with Java bean objects in JSPs, such as to access bean values. • It is a collection of utility tags that provides convenient access for interacting. • The remaining tags are used to render objects to the JSP output. • Most of the tags are used to capture references to specific objects and store them in JSP scripting variables.	• It is used to implement simple conditional logic in JSP. • It provides a rich set of tags for cleanly implementing simple conditional logic in JSP. • You can wrap content that will be processed only when a particular condition is true.

JSP Tag Libraries 4-4

- Some of the logic tags are as follows:
  - present
  - notPresent
  - Equal
  - notEqual
  - greaterThan
  - lessThan
  - greaterEqual
  - lessEqual

Developing View 1-2

- Following code snippet form shows login.jsp page containing one text field to get the user name and one password field to get the password:

```
<%@taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<html>
<head>
<title>Login Page</title>
</head>
<body>
<div style="color:red">
<html:errors />
</div>
<html:form action="/Login" >
User Name :<html:text name="LoginForm" property="userName" />
Password :<html:password name="LoginForm"property="password" />
<html:submit value="Login" />
</html:form> </body> </html>
```

- The action is `/Login`, the input page is `login.jsp` and the corresponding action class is `LoginAction.java`.

Developing View 2-2

- Following code snippet shows the `success.jsp` and `failure.jsp` pages:

```
<!—success.jsp -->
<html>
. . .
<body>
        <h1>Login Success. Welcome <bean:write name="LoginForm"
property="userName"></bean:write></h1>
    </body>
</html>
<!—failure.jsp -->
<html>
. . .
  <div style="color:red">
  <h1>Invalid user name  <bean:write name="LoginForm"
property="userName"></bean:write></h1>
 </div>
</html>
```

Summary

- The MVC is an architectural pattern whose main concern is to separate data from the user interface.
- In Model 1 architecture, the JSP page is not only responsible for processing requests and sending back replies to the clients, but also for extracting the HTTP request parameters, invoking the business logic, and handling the HTTP session.
- Model 2 architecture is an approach used for developing a Web application. It separates the 'Business Logic' from the 'Presentation Logic'. Besides this, Model 2 has an additional component - a Controller.
- The responsibility of the Controller Servlet is to process the incoming request and instantiate a Model - a Java object or a bean to compute the business logic.
- A framework provides a structural support in the form of classes that can be extended for reuse and a functional toolkit in the form of software libraries.
- Struts provides the foundation along with libraries and utilities to develop MVC based applications easily and faster.
- The major constituents of Struts framework are: base framework, JSP tag libraries, tiles plugin, and validator plugin.
- In Struts, Controller component is in the form of ActionServlet which executes Action object. Action object interacts with the Model and prepares the data for View.
- Struts come packaged with a set of its own custom JSP tag libraries that aids in the development of JSPs.