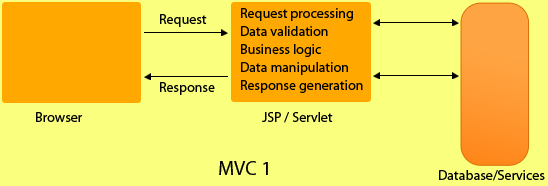
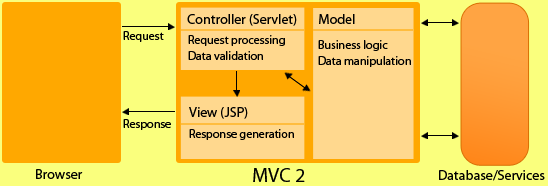
***Architectural Diagram***

**MVC 1 Architecture**



**MVC 2 Architecture**



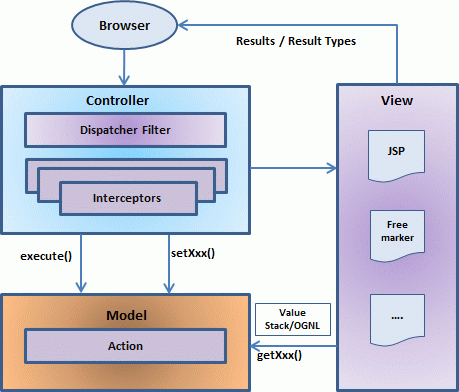
***Difference Between MVC 1 & MVC 2***

|  |  |
| --- | --- |
| **MVC  1** | **MVC 2** |
| MVC1 associates the presentation logic with the business logic. | MVC 2 isolates or disassociates the presentation logic from business logic |
| In MVC1, only one component is responsible for receiving request and sending response. | In MVC2, there is separate components for receiving and  sending response. i.e. Controller & View. |
| In MVC 1, business logic and presentation Logic is combined so web designer and web developer cant work simultaneously. | Since both logics are separate that's why designer and developer can work together. |
| Doesn't support reusability of application components. | Reusability of components |
| In MVC 1, controller and model, both are JSP. | While controller is servlet and model is java class. |
| In MVC1 there is tight coupling between page and model as data access is usually done using Custom tag or through java bean call. | In MVC2 architecture there is only one controller which receives all the request for the application and is responsible for taking appropriate action in response to each request. |

From a high level, Struts2 is a pull-MVC (or MVC2) framework. The Model-View-Controller pattern in Struts2 is realized with following five core components:

* **Actions**
* **Interceptors**
* **Value Stack / OGNL**
* **Results / Result types**
* **View technologies**

Struts 2 is slightly different from a traditional MVC framework in that the action takes the role of the model rather than the controller, although there is some overlap.



The above diagram depicts the **M**odel, **V**iew and **C**ontroller to the Struts2 high level architecture. The controller is implemented with a Struts2 dispatch servlet filter as well as interceptors, the model is implemented with actions, and the view as a combination of result types and results. The value stack and OGNL provide common thread, linking and enabling integration between the other components.

Apart from the above components, there will be a lot of information that relates to configuration. Configuration for the web application, as well as configuration for actions, interceptors, results, etc.

This is the architectural overview of the Struts 2 MVC pattern. We will go through each component in more detail in the subsequent chapters.

Request life cycle:

Based on the above digram, one can explain the user's request life cycle in Struts 2 as follows:

* User sends a request to the server for requesting for some resource (i.e pages).
* The FilterDispatcher looks at the request and then determines the appropriate Action.
* Configured interceptors functionalities applies such as validation, file upload etc.
* Selected action is executed to perform the requested operation.
* Again, configured interceptors are applied to do any post-processing if required.
* Finally the result is prepared by the view and returns the result to the user.

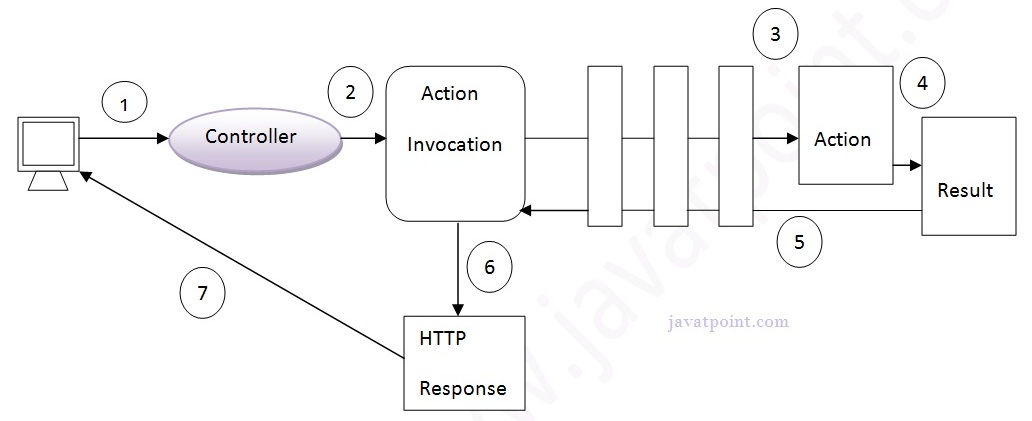
|  |  |
| --- | --- |
| 1 | **Action** Create an action class which will contain complete business logic and control the interaction between the user, the model, and the view. |
| 2 | **Interceptors** Interceptor is an object that is invoked at the preprocessing and post processing of a request. In Struts 2, interceptor is used to perform operations such as validation, exception handling, internationalization, displaying intermediate result etc. Create interceptors if required, or use existing interceptors. This is part of Controller. |
| 3 | **View** Create a JSPs to interact with the user to take input and to present the final messages. |
| 4 | **Configuration Files** Create configuration files to couple the Action, View and Controllers. These files are struts.xml, web.xml, struts.properties. |
| 5 | **Value Stack:** A valueStack is simply a stack that contains application specific objects such as action objects and other model object. At the execution time, action is placed on the top of the stack. We can put objects in the valuestack, query it and delete it. |
| 6 | **ActionContext:** The ActionContext is a container of objects in which action is executed. The values stored in the ActionContext are unique per thread (i.e. ThreadLocal). So we don't need to make our action thread safe.  We can get the reference of ActionContext by calling the getContext() method of ActionContext class. It is a static factory method. For example:  ActionContext context = ActionContext.getContext(); |
| 7 | **Action Invocation:** The ActionInvocation represents the execution state of an action. It holds the action and interceptors objects.  **ActionInvocation Interface:** The struts framework provides ActionInvocation interface to deal with ActionInvocation. It provides many methods, some of them can be used to get the instance of ValueStack, ActionProxy, ActionContext, Result etc. |
| 8 | **OGNL:** The Object Graph Navigation Language (OGNL) is an expression language. It simplifies the accessibility of data stored in the ActionContext.  The struts framework sets the ValueStack as the root object of OGNL. Notice that action object is pushed into the ValueStack. We can direct access the action property. <s:property value="username"/>  Here, username is the property key.  The struts framework places other objects in ActionContext also e.g. map representing the request, session, application scopes.  To get these values i.e. not the action property, we need to use # notation. For example to get the data from session scope, we need to use #session as given in the following example:  <s:property name="#session.username"/> (or) <s:property name="#session['username']"/> |
|  |  |

The **architecture and flow of struts 2 application**, is combined with many components such as Controller, ActionProxy, ActionMapper, Configuration Manager, ActionInvocation, Inerceptor, Action, Result etc.

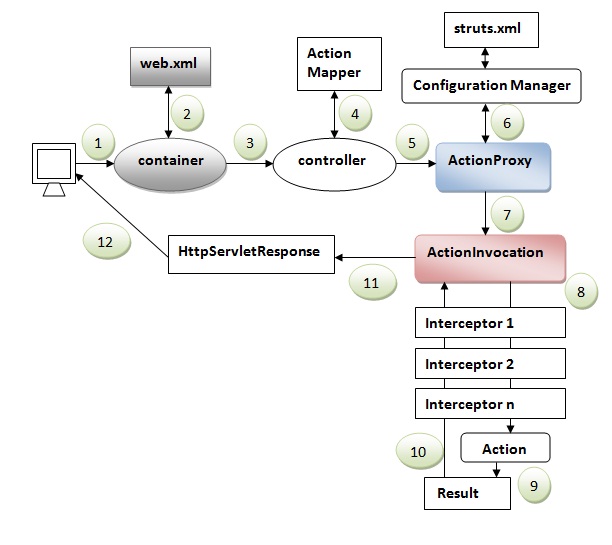
Here, we are going to understand the struts flow by 2 ways:

1. struts 2 basic flow
2. struts 2 standard architecture and flow provided by apache struts

#### Struts 2 basic flow



#### Struts 2 standard flow (Struts 2 architecture)



1. User sends a request for the action
2. Container maps the request in the web.xml file and gets the class name of controller.
3. Container invokes the controller (StrutsPrepareAndExecuteFilter or FilterDispatcher). Since struts2.1, it is StrutsPrepareAndExecuteFilter. Before 2.1 it was FilterDispatcher.
4. Controller gets the information for the action from the ActionMapper
5. Controller invokes the ActionProxy
6. ActionProxy gets the information of action and interceptor stack from the configuration manager which gets the information from the struts.xml file.
7. ActionProxy forwards the request to the ActionInvocation
8. ActionInvocation invokes each interceptors and action
9. A result is generated
10. The result is sent back to the ActionInvocation
11. A HttpServletResponse is generated
12. Response is sent to the user