Execution flow of struts

- When a client request is given, a web container will receive request
- Web container loads web.xml and verifies whether the url-patterns are verified or not, if matches web-container transfer the request to FilterDispatcher
- FilterDispatcher hand overs the request to ActionProxy, it is a proxy class which is responsible to apply before and after services to original business logic
- ActionProxy contacts ConfiguraionManager class, to know the suitable Action for the request and the needed services for the request
- ConfigurationManager class loads structs.xml and provides the required information back to ActionProxy
- ActionPorxy delegates the request along with its information to ActionInvocation
- ActionInvocation executes the interceptors added to an Action from 1 – N, after that it will call the business logic implemented from N – 1 in reverse order
- ActionInvocation receives finally result produced by an action aclass
- ActionProxy transfers the result back to FilterDispatcher
- FilterDispatcher selects an appropriate view, basing on the result
- Finally FilterDispatcher uses RequestDispatchers forwarding mechanism and forward a view as a response back to the client

Struts 2 Basic Flow

The basic flow of a Struts 2 application involves several key components working together to handle user requests and generate responses. Here's a simplified overview of the basic flow in a Struts 2 application:

The Basic Struts 2 Flow is as follows:

- **User Initiates a Request:** The user interacts with the application, typically by submitting a form or clicking on a link. The user's action triggers an HTTP request.
- **Request Reaches the Controller:** The request is received by the Struts 2 controller, which acts as an entry point for all requests. The controller is responsible for dispatching the request to the appropriate action.

- **Action Mapping:** The controller examines the request and determines which action should handle it. This mapping is usually defined in a configuration file, such as struts.xml.
- **Action Processing:** The controller instantiates the appropriate action class based on the mapping and calls its methods to process the request. Actions in Struts 2 are POJOs (Plain Old Java Objects) that contain the business logic related to the request.
- **Parameter Population:** When the controller creates the action instance, it populates the action's properties with data from the request parameters. Struts 2 uses the built-in type conversion mechanism to convert the incoming string values into appropriate Java types.
- **Execution of the Business Logic:** The action class executes the business logic associated with the request. This may involve interacting with a database, performing calculations, or invoking other services.
- **Result Generation:** After the business logic is executed, the action returns a result code to the controller. The resulting code represents the next step to be taken.
- **Result Processing:** The controller uses the result code to determine which result type should be applied. The result type is configured in the struts.xml file and specifies how the response should be generated.
- **Result Execution:** The selected result type processes the result and generates the response. This can include rendering a JSP page, generating JSON or XML, redirecting to another URL, or performing other actions.
- **Response Sent to the User:** The generated response is sent back to the user's browser, completing the request-response cycle.

It's important to note that this is a simplified overview, and the actual flow can vary depending on the specific configuration and customization of your Struts 2 application. Additionally, Struts 2 provides a rich set of features and additional components, such as interceptors and form validation, which can further enhance the application flow.

Struts 2 Standard Flow or Struts 2 Architecture

The standard flow in a Struts 2 application follows the Model-View-Controller (MVC) architectural pattern. Here's an overview of the standard flow in a Struts 2 application:

The Standard Struts 2 Flow is as follows:

1. **User Initiates a Request:** The user interacts with the application, typically by submitting a form or clicking on a link. The user's action triggers an HTTP request.
2. **Request Reaches the Front Controller:** The request is received by the Struts 2 front controller, known as the "ActionServlet." The ActionServlet acts as the central entry point for all requests in a Struts 2 application.
3. **Action Mapping:** The ActionServlet examines the request and determines which action should handle it based on the configured action mappings. The action mapping defines the relationship between a URL pattern and the corresponding action class.
4. **Action Creation and Execution:** Once the ActionServlet identifies the appropriate action class, it creates an instance of that class and executes it. The action class implements the business logic related to the request.
5. **Parameter Population:** Before executing the action method, Struts 2 populates the action's properties with data from the request parameters. Struts 2 uses its built-in type conversion mechanism to convert the incoming string values into appropriate Java types.
6. **Validation (optional):** If the action class specifies validation rules using annotations or XML configuration, Struts 2 performs automatic validation. The validation checks the submitted data against the specified rules and collects any validation errors.
7. **Execution Of Business Logic:** The action method is invoked, and the action class executes the business logic associated with the request. This may involve interacting with a database, performing calculations, or invoking other services.
8. **Result Generation:** After the business logic is executed, the action returns a result code to the ActionServlet. The result code represents the next step to be taken.
9. **Result Processing:** The ActionServlet uses the result code to determine which result type should be applied. The result type is configured in the struts.xml file and specifies how the response should be generated.
10. **Result Execution:** The selected result type processes the result and generates the response. This can include rendering a JSP page, generating JSON or XML, redirecting to another URL, or performing other actions.

11. **Response Sent to the User:** The generated response is sent back to the user's browser, completing the request-response cycle.

Throughout this flow, Struts 2 provides various features and components, such as interceptors, form validation, and result types, which allow for flexibility and customization in handling different types of requests and generating responses.

How Struts 2 Works?

Struts 2 is a popular Java web application framework that follows the Model-View-Controller (MVC) architectural pattern. It provides a structured approach to building web applications by separating the concerns of handling user requests, processing business logic, and generating responses. Here's a high-level explanation of how Struts 2 works:

1. **Configuration:** Struts 2 applications are typically configured using XML files, such as struts.xml, which define various components, mappings, and settings. Configuration files specify things like action mappings, result types, interceptors, and global settings.
2. **Front Controller:** The heart of a Struts 2 application is the front controller, known as the "ActionServlet." It acts as the central entry point for all requests and handles the overall flow of the application. When a request is received, the ActionServlet examines the URL and delegates it to the appropriate action for processing.
3. **Actions:** Actions in Struts 2 are Java classes that implement the business logic for handling specific user requests. Each action typically corresponds to a specific functionality or page in the application. Actions are simple Plain Old Java Objects (POJOs) and are typically designed to be stateless.
4. **Action Mapping:** The ActionServlet uses the configuration file to determine the appropriate action to handle a request based on the URL. The action mapping specifies the relationship between the URL pattern and the corresponding action class.
5. **Parameter Population:** Before executing the action method, Struts 2 automatically populates the properties of the action class with data from the request parameters. It uses its built-in type conversion mechanism to convert the incoming string values to the appropriate Java types.
6. **Business Logic:** The action class contains the business logic for processing the request. It can interact with databases, perform calculations, call external services, or perform any other required operations.

7. **Result Generation:** After executing the business logic, the action returns a result code to the ActionServlet. The resulting code represents the next step to be taken, such as displaying a JSP page, redirecting to another URL, or generating a response in a specific format (e.g., JSON, XML).
8. **Result Types:** Struts 2 provides a variety of result types that determine how the response is generated. Result types are configured in the XML file and specify the rendering process, such as JSP-based views, JSON or XML serialization, or even custom result types.
9. **Interceptors:** Interceptors are a key feature of Struts 2 and allow for pre-and post-processing of requests and responses. Interceptors can be configured globally or on a per-action basis to perform tasks such as authentication, logging, validation, or modifying the behavior of the request/response flow.
10. **View Rendering:** Once the result is determined, Struts 2 generates the appropriate response based on the selected result type. This may involve rendering a JSP page, generating JSON or XML, or performing other actions specified by the result type.
11. **Response Sent to User:** The generated response is sent back to the user's browser, completing the request-response cycle.

Struts 2 provides a well-defined structure and a set of features that help developers build scalable and maintainable web applications. It promotes separation of concerns, modular design, and easy extensibility through its various components and configuration options.

The explanation of the above diagram is as follows:

- **Request:** This is the very first step, using a web browser the client makes a request for an individual resource, which is then sent by the web container. After that, the **web container** loads web.xml and confirms whether the URL patterns matched or not. Once the verification is verified, the web container transfers the request to **Filter Dispatcher**.
- **StrutsPrepareAndExecuteFilter:** Once the request is sent to the filter dispatcher estimate the request and check an appropriate action as per the mapping of the URL (**ActionMapper**) calls the **ActionProxy** then reads the configuration file manager (like the struts.xml file) to check an exact action for the request. ActionProxy which reads then creates an **ActionInvocation**, accounts for the execution of command pattern implementation then the request is sent to the proper Action Class.

- **Interceptor Stacks:** Before receiving the **Action Class**, the request passes through the Interceptor Stacks, where the list of interceptors is identified which are essential to be checked before creating the Action class.
- **Action Class:** then the request that passes through the Action class, which then accomplishes the code and finally generates the result of execution as Success or Input or Error.
- **Result:** Depending on the code that is being resulted, the **Controller** finds View to be and hands over the result of Action. At the time of handling, **Struts tags** are furnished by the framework and can be used by the templates.
- **Interceptors Stack:** Before calling the client, the interceptors are to be checked again and the response returns to the user through the filters that are being configured in the web.xml.

Struts 2 Life Cycle

The life cycle of a Struts 2 application involves several stages, from initialization to request processing and response generation. Here's an overview of the life cycle stages in a Struts 2 application:

1. **Initialization:** When the web application starts, the Struts 2 framework initializes its components, such as the ActionServlet and other configuration settings. This initialization typically occurs when the web server or container starts up.
2. **Request Handling:** When a user initiates a request by accessing a URL, the request is intercepted by the front controller, which is the ActionServlet in Struts 2. The front controller is responsible for handling all incoming requests in the application.
3. **Pre-Processing:** Before the request is passed to the appropriate action, Struts 2 applies any configured interceptors. Interceptors can perform tasks such as authentication, logging, input validation, or modifying the behavior of the request. Interceptors are executed in the order they are defined in the configuration.
4. **Action Selection:** The front controller examines the URL and determines the appropriate action class based on the configured action mappings. The action mapping specifies the relationship between the URL pattern and the corresponding action class.

5. **Action Execution:** Once the appropriate action class is determined, an instance of the action class is created, and the ActionServlet calls the action's methods to execute the business logic associated with the request. The action method typically performs tasks such as data retrieval, manipulation, or interaction with other services.
6. **Result Generation:** After the action method is executed, the action returns a result code to the ActionServlet. The resulting code represents the next step to be taken. The ActionServlet uses the result code to determine the appropriate result type and execute it.
7. **Result Processing:** The selected result type processes the result and generates the response. This can involve rendering a JSP page, generating JSON or XML, redirecting to another URL, or performing other actions specified by the result type.
8. **Post-Processing:** After the response is generated, Struts 2 applies any configured interceptors for post-processing. These interceptors can perform tasks such as logging, caching, or modifying the response before it is sent back to the user.
9. **Response Sent:** The generated response is sent back to the user's browser, completing the request-response cycle.
10. **Cleanup:** Once the response is sent, any necessary cleanup tasks are performed, such as closing resources or releasing memory.

It's important to note that the life cycle stages can vary depending on the specific configuration and customization of your Struts 2 application. Struts 2 provides flexibility through its interceptor mechanism, allowing developers to customize the request and response handling process at various stages of the life cycle.

The flow is as follows:

- **User Sends request:** The user calls a request for the unspecified resource to the Servlet Container.
- **FilterDispatcher determines the appropriate action:** The FilterDispatcher accepts the request and checks the exact action corresponding to the request.
- **Interceptors are applied:** Interceptors accomplish the tasks i.e. workflow, validation, file upload, and automatically applied to the request.

- **Execution of Action:** Then the action method is responsible to perform database-related operations such as retrieving data or storing data in a database.
- **Output rendering:** The Result is generated and rendered in the view.
- **Return of Request:** The request returns through the interceptors in reverse order. It gives permission to perform the clean-up or additional processing.
- **Display the result to the user:** The control is returned to the servlet container which sends the output to the user's browser.