DispatcherServlet

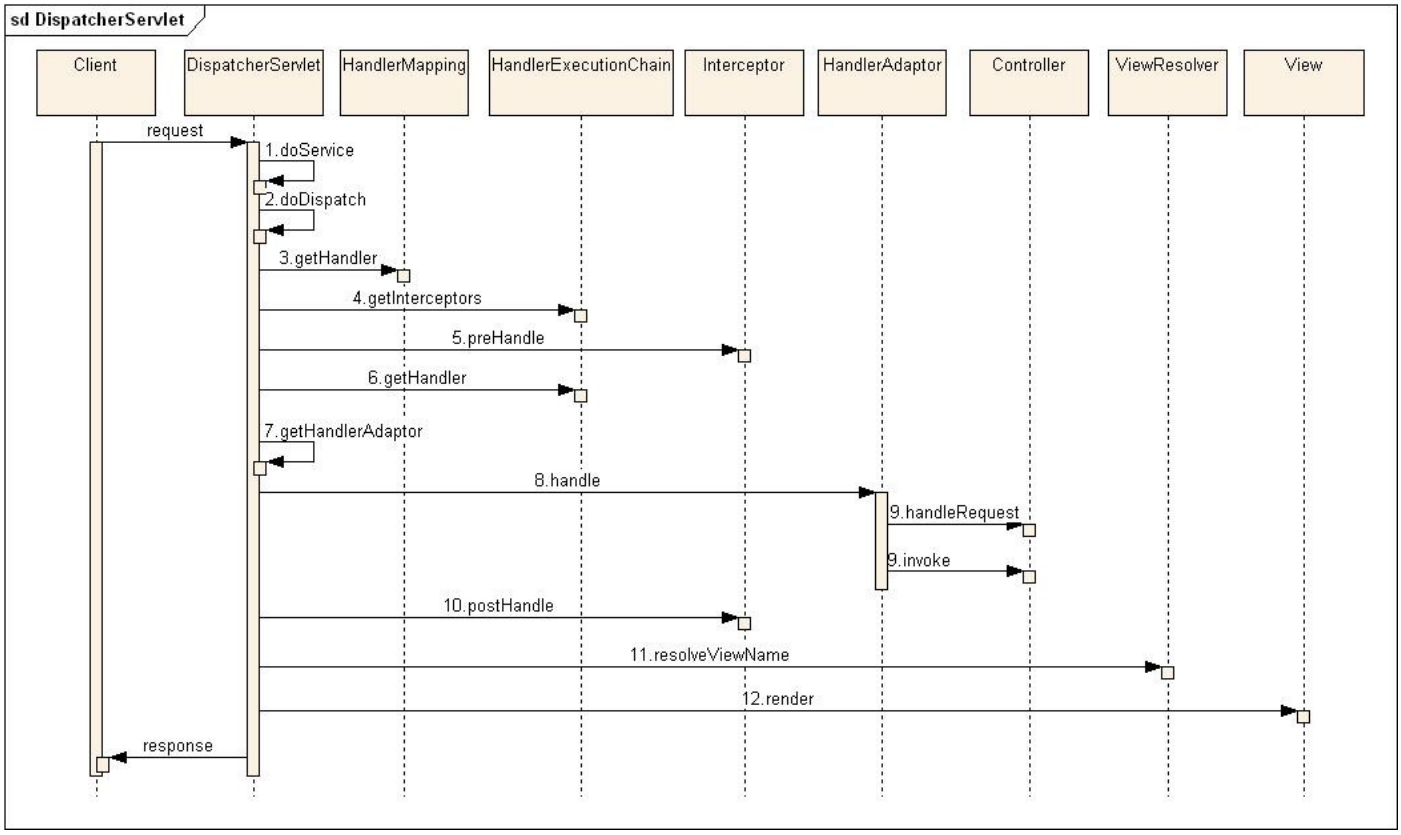
Summary

The only FrontController of the Spring MVC Framework is the key element of Spring MVC. DispatcherServlet is the approach point of any web request which processes the web request and responds the result data to the client. In other words, DispatcherServlet supervise the web request life cycle of the Spring MVC.

Description

Web request flow in DispatcherServlet

The flow in the DispatcherServlet in web request from the client is as following. For more details, it is recommended to detect the process in the debugging mode.



1. The processing of web request starts from the doService method. After putting several data used in the DispatcherServlet to the request object, call the doDispatch.
2. The tasks from no.3 to 13 is within the doDispatch method. The real web request processing is processed using components such as controller and view.
3. getHandler method obtains the request controller using RequestMapping object.
4. When you find the request handler, the HandlerExecutionChain should include request interceptors in returning the HandlerExecutionChain object.
5. If there are interceptors to be run, the preHandle method of the interceptor should be run respectively.
6. The instance of the controller should be obtained using getHandler of the HandlerExecutionChain.
7. Many HanlderAdaptors can be configured as HandlerMapping. The getHandlerAdaptor method returns one appropriate HanlderAdaptor to the controller.
8. The handle method of selected HanlderAdaptor is executed. The actual execution runs the controller handled by the parameter.
9. For hierarchal controllers, the handleRequest method is executed. For @Controller, the HanlderAdaptor(AnnotationMethodHandlerAdapter) invoke() the controller method using HandlerMethodInvoker.
10. postHandle method of the interceptor is run.

11. resolveViewName method returns the view object with the logical view name.
12. To show the data of model object, the render method of the view object is operated.

Configuring DispatcherServlet in web.xml

To use the Spring MVC Framework, the DispatcherServlet should be configured on web.xml and the files with bean data should be configured to create WebApplicationContext.

Basic configuration

```
<web-app>
        <!-- DispatcherServletprocesses the web request of the easycompnay web application.-->
        <servlet>
                <servlet-name>easycompany</servlet-name>
                <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
         </servlet>
</web-app>
```

The servlet-name becomes a prefix of the bean configufation file which will be referred as default. It is the same configuration as (servlet-name)-servlet.xml.
If the web.xml is written as above example, the DispatcherServlet searches/WEB-INF/easycompany-servlet.xml as default.

Configuration using contextConfigLocation

When you want to use more than one bean configuration files or designate the file name and route, set the bean configuration file route on the initialization parameter value, contextConfigLocation.

```
...
        <servlet>
                <servlet-name>easycompany</servlet-name>
                <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
                <init-param>
                        <param-name>contextConfigLocation</param-name>
                        <param-value>
                         /WEB-INF/config/easycompany-web.xml
                        </param-value>
                </init-param>
         </servlet>
```

Configuration using ContextLoaderListener

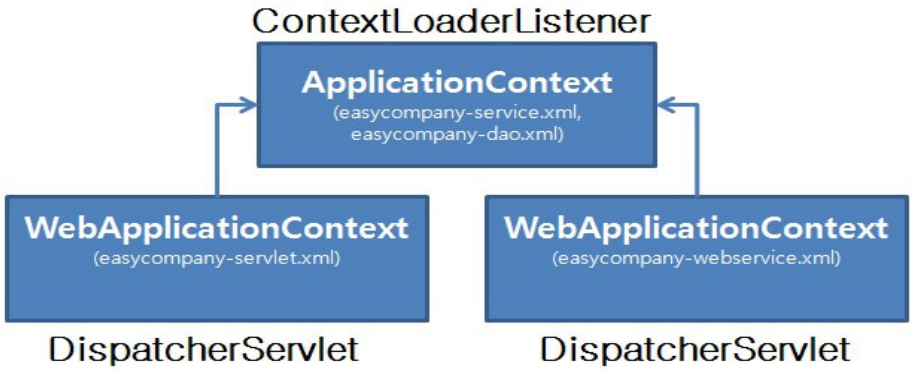
Generally, the bean configuration file is divided into layers of persistence, service and web rather than used as one file. Additionally, in some cases, DispatcherServlet uses more than two beans of the same persistance, service layer
In this case, the configuration data of the common bean (persistance, service) is recommended in ApplicationContext and the beans in the web layerare recommended as below which stores in WebApplicationContext.
The common bean configuration file lodas the org.springframework.web.context.ContextLoaderListener which is registered as the sublet listener to create the ApplicationContext, and the bean configuration file of the web later loads the DispatcherServlet to crate the WebApplicationContext.

```
....
        <!-- ApplicationContext bean configuration file-->
        <context-param>
                <param-name>contextConfigLocation</param-name>
                <param-value>
                        <!—The bean configuration file are classified with enter, commas and
semicolons.-->
```

```
                    /WEB-INF/config/easycompany-service.xml,/WEB-
INF/config/easycompany-dao.xml
                </param-value>
        </context-param>

        <!-- It loads the ApplicationContext in the starting point of web application and the loaded
bean data is referred from all WebApplicationContext. -->
        <listener>
                <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
        </listener>

        <servlet>
                <servlet-name>employee</servlet-name>
                <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
                <init-param>
                        <param-name>contextConfigLocation</param-name>
                        <param-value>
                                /WEB-INF/config/easycompany-service.xml
                        </param-value>
                </init-param>
        </servlet>

        <servlet>
                <servlet-name>webservice</servlet-name>
                <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
                <init-param>
                        <param-name>contextConfigLocation</param-name>
                        <param-value>
                                /WEB-INF/config/easycompany-webservice.xml
                        </param-value>
                </init-param>
        </servlet>
....
```

The bean data of this ApplicationContext is referred by any WebApplicationContext. For instance, even though the DispatcherServlet uses two, when the web.xml that uses the service, DAO as below, the bean data declared in easycompany-servlet.xml cannot be referred in easycompany-webservice.xml. However, the bean data configured in easycompany-service.xml, easycompany-dao.xml refers to easycompany-servlet.xml and easycompany-webservice.xml.

The relations of ApplicationContext and WebApplicationContext are as following:



Reference

- The Spring Framework - Reference Documentation 2.5.6
- Spring Framework API Documentation 2.5.6