**Spring Interceptor**

Spring Interceptors has the ability to pre-handle and post-handle the web requests. Each interceptor class should extend the HandlerInterceptorAdapter class. Here we will create a Logger Interceptor by extending the HandlerInterceptorAdapter class. You can override any of the three callback methods preHandle(), postHandle()and afterCompletion(). As the names indicate the preHandle() method will be called before handling the request, the postHandle() method will be called after handling the request and the afterCompletion() method will be called after rendering the view.

In each method we will log information using log4j. First instantiate the logger in the static context, then set up the basic configuration so that the log messages will be logged on the console.

The LoggerInterceptor class is shown below.

01.package com.tutorials4u.interceptor;

02.

03.import javax.servlet.http.HttpServletRequest;

04.import javax.servlet.http.HttpServletResponse;

05.

06.import org.apache.log4j.BasicConfigurator;

07.import org.apache.log4j.Logger;

08.import org.springframework.web.servlet.ModelAndView;

09.import org.springframework.web.servlet.handler. HandlerInterceptorAdapter;

10.

11.public class LoggerInterceptor extends HandlerInterceptorAdapter {

12.

13.static Logger logger = Logger.getLogger(LoggerInterceptor.class);

14.

15.static{

16.BasicConfigurator.configure();

17.}

18.

19.@Override

20.public boolean preHandle(HttpServletRequest request,

21.HttpServletResponse response, Object handler) throws Exception {

22.logger.info("Before handling the request");

23.return super.preHandle(request, response, handler);

24.}

25.

26.@Override

27.public void postHandle(HttpServletRequest request,

28.HttpServletResponse response, Object handler,

29.ModelAndView modelAndView) throws Exception {

30.logger.info("After handling the request");

31.super.postHandle(request, response, handler, modelAndView);

32.}

33.

34.@Override

35.public void afterCompletion(HttpServletRequest request,

36.HttpServletResponse response, Object handler, Exception ex)

37.throws Exception {

38.logger.info("After rendering the view");

39.super.afterCompletion(request, response, handler, ex);

40.}

41.}

Now the logger interceptor is created you need to associate this interceptor with the handler mapping. Here we use BeanNameUrlHandlerMapping, incase you are using more than one handler mapping you need to associate the interceptor with each one of them. The code below shows how to associate an interceptor with the handler mapping.

01.<?xml version="1.0" encoding="UTF-8"?>

02.<beans xmlns="http://www.springframework.org/schema/beans"

03.xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

04.xmlns:p="http://www.springframework.org/schema/p"

05.xsi:schemaLocation="http://www.springframework.org/schema/beans

06.http://www.springframework.org/schema/beans/spring-beans.xsd">

07.

08.<bean id="viewResolver" class="org.springframework.web.servlet.view. InternalResourceViewResolver" p:prefix="/WEB-INF/jsp/"p:suffix=".jsp" />

09.

10.<bean id="handlerMapping" class="org.springframework.web.servlet.handler. BeanNameUrlHandlerMapping" p:interceptors-ref="loggerInterceptor" />

11.

12.<bean id="loggerInterceptor" class="com.tutorials4u.interceptor.LoggerInterceptor" />

13.

14.

15.<bean id="userService" class="com.tutorials4u.service.UserServiceImpl" />

16.

17.<bean name="/userRegistration.htm" class="com.tutorials4u.web.UserController" p:userService-ref="userService"p:formView="userForm" p:successView="userSuccess" />

18.

19.</beans>

**Spring Interceptor Using Annotation Tutorial**

If you are using annotated Spring controllers, the only change that you need to do to the previous interceptor example is the configuration. In the Spring bean configuration file instead of using BeanNameUrlHandlerMapping or any other handler mapping use DefaultAnnotationHandlerMapping. The configuration file is shown below.

01.<?xml version="1.0" encoding="UTF-8"?>

02.

03.<beans xmlns="http://www.springframework.org/schema/beans"

04.xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"

05.xmlns:context="http://www.springframework.org/schema/context"

06.xsi:schemaLocation="http://www.springframework.org/schema/beans

07.http://www.springframework.org/schema/beans/spring-beans.xsd

08.http://www.springframework.org/schema/context

09.http://www.springframework.org/schema/context/spring-context.xsd">

10.

11.<bean id="viewResolver" class="org.springframework.web.servlet.view. InternalResourceViewResolver" p:prefix="/WEB-INF/jsp/"p:suffix=".jsp" />

12.

13.<bean class="org.springframework.web.servlet.mvc.annotation. DefaultAnnotationHandlerMapping" p:interceptors-ref="loggerInterceptor" />

14.

15.<context:component-scan base-package="com.tutorials4u.web" />

16.

17.<bean id="loggerInterceptor" class="com.tutorials4u.interceptor.LoggerInterceptor" />

18.

19.<bean id="userService" class="com.tutorials4u.service.UserServiceImpl" />

20.

21.

22.</beans>