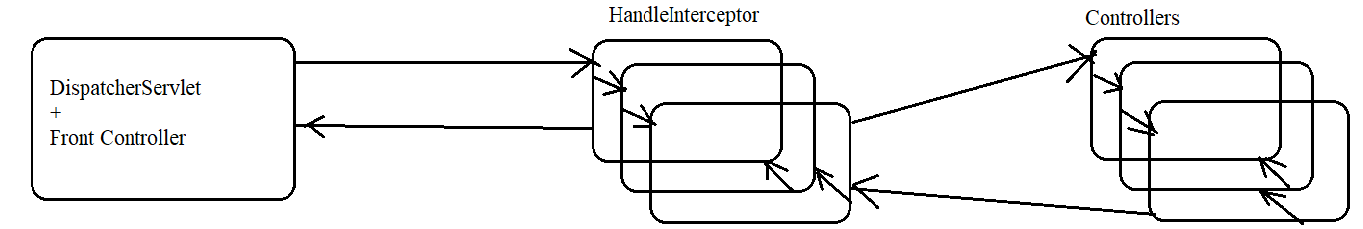
# Handler Interceptors

These are extension hooks to Controller/handler classes to add pre/post logics having ability to enable or disable without touching the source code of Controller/handler class.



HandleInterceptor Methods are

Default void afterCompletetion(req, res, Object Hander, Exception ex) throws Exception

Executes after rendering the view component. It is useful for clean up operations like removing data from session scopes

Default void postHandle(req, res, Object handler, ModealAndView mav) throes Exception

Executes after executing the controller. It is useful for post processing logics like converting model attributes values to different formats like changing number of formats, currency symbols, date formats and etc..

Default Boolean perHandle(req, res, Object handler) throws Exception

Executes before executing the controller. It is useful for pre processing logics like checking browser type, checking timings , authentication and etc… if this method returns “true” then Controller flow of execution will continue, otherwise the controller eill not execute.

Up to Spring 4 developer preferred developing hanler/ controller class by extending from HandlerInterceptorAdapter class which is a pre-defined class implementing HandleInterceptor(I) providing null method definitions for all the methods, so we can override only those in which they are interested in. This has been changed in spring 5, we can develop interceptor class directly by implementing HandleInterceptor(I) because it is given java 8 interface default methods.

How to Apply specific interceptors on specific controller classes ?

Map related controller classes with their incoming request uri using separate Handler mapping and link interceptor on controller classes using that handler mapping.

<bean

class=*"org.springframework.web.servlet.handler.SimpleUrlHandlerMapping"*>

<property name=*"mappings"*>

<props>

<prop key=*"welcome.htm"*>shc</prop>

</props>

</property>

<property name=*"interceptoes"*>

<list>

<bean class=*"learn.luv2code.TimechekcingInteceptor"*/>

</list>

</property>

</bean>

# View Resolvers:

There are given to resolve/identify the physical view component name and location based on the given logical view name. It returns view object having the name and location of physical view component. All View Resolver are implementation classes of ViewResolver(I). View objects are the objects of classes implementing View(I) on this View object render(-) method weill be called in order to pass the control to physical UI component.

URLBaseVIewReoslver:

Capable of resolving/identifying physical view name and location of any technology, but we must configure View class explicitly. For locating JSP/Servlet component in private area configure InternalResourceView or Jstlview.

What is difference between JStlVIew and InternalResourceView classes?

InternalResourceView makes the programmer to add Jstl jar files

to WEB-INF/lib folder only when jstl tags are used in the jsp pages otherwise not required.

JstlView makes the programmer to add Jstl jar files to WEB-INF/lib folder only whether jstl tags are used or not in the jsp pages otherwise not required.

ReourceBundleViewResolver

Allows to configure view class name, url separately for each logical view name by taking the support od properties file. It allows to configure different technology view components for different logical view names and also allows to place different view components in different locations with different extensions. Default properties file name is views.properties in src folder, If we want to specify other location and name then we need to specify these details explicitly.