HandlerMapping

Summary

When there is http request from the client on the DispatcherServlet, the HandlerMapping maps the controller to process the request.
Spring MVC has the implementation class of the interface, many number of HandlerMapping can be used by purpose.
When there is no definition on HandlerMapping in bean definition file, Spring MVC uses default HandlerMapping.
Default HandlerMapping is BeanNameUrlHandlerMapping and in upper environment than jdk1.5, DefaultAnnotationHandlerMapping is also HandlerMapping.

Description

The implementation class of major HandlerMapping such as BeanNameUrlHandlerMapping and SimpleUrlHandlerMapping extends the upper abstract class, AbstractHandlerMapping and AbstractUrlHandlerMapping. Therefore, these abstract classes use properties.
Main properties are as follows:

- defaultHandler: If there is no controller corresponding to the request, return the controller registered in defaultHandler.
- alwaysUseFullPath: Indicate whether to use URL full path during URL and Controller mapping. For example, if servlet-mapping is /easycompany/* and alwaysUseFullPath is true, /easycompany/employeeList.do, if alwaysUseFullPath is false, /employeeList.do.
- interceptors: register interceptor to perform specific logic before and after Controller processes request. Several interpceptors can be registered. See here for details on interceptor.
- order: priority when using several HandlerMapping.

```
...
<bean class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping" p:order="2"/>
<bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping" p:order="3">
...
```

- pathMatcher: register pathMatcher supporting the matching of specific style when matching user request URL path and URL path of setting information. Default value is AntPathMatcher providing the pattern matching of Ant-style.

Main HandlerMapping implementation classes which the Spring MVC provide are as below:

- BeanNameUrlHandlerMapping
- ControllerClassNameHandlerMapping
- SimpleUrlHandlerMapping
- DefaultAnnotationHandlerMapping

Since DefaultAnnotationHandlerMapping cannot apply interceptor in the URL unit at @MVC, e-gov framework added the HandlerMapping implementation class as shown below.

- SimpleUrlAnnotationHandlerMapping

BeanNameUrlHandlerMapping

The BeanNameUrlHandlerMapping operates by mapping the URL declared in ***name*** attribute and the controller defined in ***class*** attribute within bean definition tag.

For example, if it is defined as follows:

```
<beans ...>
...
```

```
    <!—If BeanNameUrlHandlerMapping is the only one for HandlerMapping, no separate empty
definition is required for BeanNameUrlHandlerMapping.-->
    <!--<bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>-->
    <bean name="/insertEmployee.do"
class="com.easycompany.controller.InsertEmployeeController">
        ...
    </bean>
...
</beans>
```

If there is URL ~~/insertEmployee.do request from Client, InsertEmployeeController class takes charge of the request processing.

As mentioned earlier, if there is no empty definition of HandlerMapping in WAC(WebgApplicationContext), BeanNameUrlHandlerMapping is used(without separate empty definition).
However, if it is to be used with other HandlerMapping such as SimpleUrlHandlerMapping, empty definition should be performed in BeanNameUrlHandlerMapping too.

```
<beans ...>
...
    <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
       <property name="mappings">
        ....
       </property>
    </bean>

    <bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
    <bean name="/insertEmployee.do"
class="com.easycompany.controller.InsertEmployeeController">
        ....
    </bean>
...
</beans>
```

ControllerClassNameHandlerMapping

ControllerClassNameHandlerMapping maps the URL in lower case letters from the controller class name defined in beans that removed the controller.

```
<beans ..>
    ...
    <bean
class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"/>

    <bean class="com.easycompany.controller.hierarchy.EmployeeListController"/>

    <bean class="com.easycompany.controller.hierarchy.InsertEmployeeController"/>
    ...
</beans>
```

If empty definition is as shown above, url mapping is performed as shown in EmployeeListController ↔ /employeelist*, InsertEmployeeController ↔ /insertemployee*.
caseSensitive or pathPrefix, basePackage can be set as a property value in ControllerClassNameHandlerMapping.

- caseSensitive: whether to use capital letter when mapping URL path in Controller name. (ex. To be used as /easycompany/insertEmployee* not as /insertemployee*).
- pathPrefix: default prefix value in URL path. Default value is false.
- basePackage: Default package name of controller to be used in URL mapping. If there is a subpackage to be added to default package in the package name of the Controller used, name of relevant subpackage is added to URL path.

```
<beans ..>
   ...
   <bean
class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping">
      <property name="pathPrefix" value="/easycompany"/>
      <property name="caseSensitive" value="true"/>
      <property name="basePackage" value="com.easycompany.controller"/>
   </bean>

   <bean class="com.easycompany.controller.hierarchy.EmployeeListController"/>

   <bean class="com.easycompany.controller.hierarchy.InsertEmployeeController"/>
   ...
</beans>
```

Then, url mapping is performed as shown in EmployeeListController ↔ /easycompany/hierarchy/employeeList*, InsertEmployeeController ↔ /easycompany/hierarchy/insertEmployee* .

SimpleUrlHandlerMapping

SimpleUrlHandlerMapping supports the Ant-Style pattern matching and many URL's can be mapped in one controller.

Designate the URL pattern in the key value of property and designate the id or name of controller in the value.

```
<beans ...>
...
        <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
            <property name="mappings">
                <props>
                    <prop key="/employeeList.do">employeeListController</prop>
                    <prop key="/insertEmployee.do">insertEmployeeController</prop>
                    <prop key="/updateEmployee.do">updateEmployeeController</prop>
                    <prop key="/loginProcess.do">loginController</prop>
                    <prop key="/**/login.do">staticPageController</prop>
                    <prop key="/static/*.html">staticPageController</prop>
                </props>
            </property>
        </bean>

        <bean id="loginController" class="com.easycompany.controller.hierarchy.LoginController"/>
        <bean id="employeeListController"
class="com.easycompany.controller.hierarchy.EmployeeListController" />
        <bean id="insertEmployeeController"
class="com.easycompany.controller.hierarchy.InsertEmployeeController" />
        <bean id="updateEmployeeController"
class="com.easycompany.controller.hierarchy.UpdateEmployeeController" />
        <bean id="staticPageController"
class="org.springframework.web.servlet.mvc.UrlFilenameViewController" />
...
</beans>
```

Interceptor can be applied in the unit of specific URL if using SimpleUrlHandlerMapping.
Declare interceptor to be applied to property interceptors as a list.
To validate whether to authenticate the user for URL /employeeList.do, /insertEmployee.do, /updateEmployee.do request, define as shown below.

```
<beans ...>
...
        <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
```

```
                <property name="interceptors">
                    <list>
                        <ref local="authenticInterceptor"/>
                    </list>
                </property>
                <property name="mappings">
                        <props>
                                <prop key="/employeeList.do">employeeListController</prop>
                                <prop key="/insertEmployee.do">insertEmployeeController</prop>
                                <prop key="/updateEmployee.do">updateEmployeeController</prop>
                        </props>
                </property>
        </bean>
        <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
                <property name="mappings">
                        <props>
                                <prop key="/**/login.do">staticPageController</prop>
                                <prop key="/static/*.html">staticPageController</prop>
                        </props>
                </property>
        </bean>

        <bean id="loginController" class="com.easycompany.controller.hierarchy.LoginController"/>
        <bean id="employeeListController"
class="com.easycompany.controller.hierarchy.EmployeeListController" />
        <bean id="insertEmployeeController"
class="com.easycompany.controller.hierarchy.InsertEmployeeController" />
        <bean id="updateEmployeeController"
class="com.easycompany.controller.hierarchy.UpdateEmployeeController" />
        <bean id="staticPageController"
class="org.springframework.web.servlet.mvc.UrlFilenameViewController" />

        <bean id="authenticInterceptor" class="com.easycompany.interceptor.AuthenticInterceptor"
/>
...
</beans>
```

DefaultAnnotationHandlerMapping

To develop @MVC, DefaultAnnotationHandlerMapping should be used rather than aforementioned HandlerMapping in the environment above jdk 1.5.

If it is the development environment over jdk 1.5, DefaultAnnotationHandlerMapping is also default HandlerMapping along with BeanNameUrlHandlerMapping.
Accordingly, it is not required to separately declare in empty setting file(provided, however, that it should be declared if using with other HandlerMapping.)

If designating the package to perform component scan as shown below,

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:p="http://www.springframework.org/schema/p"
        xmlns:context="http://www.springframework.org/schema/context"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
                                    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-2.5.xsd">

      <context:component-scan base-package="org.mycode.controller" />

</beans>
```

Mapping the method of relevant @Controller class with URL declared in @RequestMapping among @controllers under package org.mycode.controller .
As a simple example,

```
package org.mycode.controller;
....
@Controller
public class HelloController {

	@RequestMapping(value="/hello.do")
	public String hellomethod(){
		......
	}
}
```

If URL request is received to /hello.do, method hellomethod of HelloController is executed.

SimpleUrlAnnotationHandlerMapping

When the interceptor is registered to DefaultAnnotationHandlerMapping, there is a problem which the interceptor is applied to all @Controllers.

SimpleUrlAnnotationHandlerMapping was developed to apply Interceptor in the unit of url when using @Controller.
There is someone who has similar concern and presents similar resolutions. See Scott Murphy's Blog.
Since there is no difference in terms of function, we'll introduce developed sources first.
If an alternative to apply Interceptor in url unit(method unit of Controller) from Spring Source, SimpleUrlAnnotationHandlerMapping should be deprecated as follows.
Following 3 are considered for SimpleUrlAnnotationHandlerMapping.

- HandlerMapping: Do not break the spring structure of Interceptors relationship. (ex. Controller:Interceptor (X))
- Select the method similar to existing HandlerMapping for easy use.(ex.SimpleUrlHandlerMapping)
- Perform the least customizing. → Short time… In addition, to minimize the effects to the system in case of deprecation later.

When web application is initially operated, DefaultAnnotationHandlerMapping performs 2 major works. (other HandlerMapping performs similar tasks.)

1. Read the url information of @RequestMapping and perform mapping between relevant Controller and url.
2. Read the information for Interceptor established.

No. 1 job is performed at AbstractDetectingUrlHandlerMapping, the upper class of DefaultAnnotationHandlerMapping.
determineUrlsForHandler method getting url list for mapping is declared abstract to implement at lower class.

```
public abstract class AbstractDetectingUrlHandlerMapping extends AbstractUrlHandlerMapping {
...
	protected void detectHandlers() throws BeansException {
		if (logger.isDebugEnabled()) {
			logger.debug("Looking for URL mappings in application context: " +
getApplicationContext());
		}
		String[] beanNames = (this.detectHandlersInAncestorContexts ?

	BeanFactoryUtils.beanNamesForTypeIncludingAncestors(getApplicationContext(),
Object.class):
					getApplicationContext().getBeanNamesForType(Object.class));
```

```
                // Take any bean name that we can determine URLs for.
                for (int i = 0; i < beanNames.length; i++) {
                        String beanName = beanNames[i];
                        String[] urls = determineUrlsForHandler(beanName);
                        if (!ObjectUtils.isEmpty(urls)) {
                                // URL paths found: Let's consider it a handler.
                                registerHandler(urls, beanName);
                        }
                        else {
                                if (logger.isDebugEnabled()) {
                                        logger.debug("Rejected bean name '" + beanNames[i] +
"': no URL paths identified");
                                }
                        }
                }
        }
        protected abstract String[] determineUrlsForHandler(String beanName);
}
```

Since determineUrlsForHandler method of DefaultAnnotationHandlerMapping brings all url lists of @RequestMapping, implement determineUrlsForHandler method to bring url list defined in empty setting file only in SimpleUrlAnnotationHandlerMapping.

```
package egovframework.rte.ptl.mvc.handler;
...
public class SimpleUrlAnnotationHandlerMapping extends DefaultAnnotationHandlerMapping {

        //contain in Set object by not allowing url list, duplicate value.
        private Set<String> urls;

        public void setUrls(Set<String> urls) {
                this.urls = urls;
        }

        /**
         * Among the url declared to @RequestMapping, return after remapping the url defined in
property urls only
         * PathMatcher is used during url mapping. If there is no PathMatcher separately registered,
use AntPathMatcher.
         * url list declared as @param urlsArray - @RequestMapping
         * return after filtering url set in @return urlsArray.
         */
        private String[] remappingUrls(String[] urlsArray){

                if(urlsArray==null){
                        return null;
                }

                ArrayList<String> remappedUrls = new ArrayList<String>();

                for(Iterator<String> it = this.urls.iterator(); it.hasNext();){
                        String urlPattern = (String) it.next();
                        for(int i=0;i<urlsArray.length;i++){
                                if(getPathMatcher().matchStart(urlPattern, urlsArray[i])){
                                        remappedUrls.add(urlsArray[i]);
                                }
                        }
                }

                return (String[]) remappedUrls.toArray(new String[remappedUrls.size()]);
        }
```

```
	/**
	 * To filter url declared as @RequestMapping
	 * org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping
	 * override method protected String[] determineUrlsForHandler(String beanName).
	 *
	 * @param beanName - the name of the candidate bean
	 * URL list corresponding to @return bean
	 */
	protected String[] determineUrlsForHandler(String beanName) {
		return remappingUrls(super.determineUrlsForHandler(beanName));
	}
}
```

Declare the url to be applied with interceptor in property urls and pattern matching of Ant-style is supported.
SimpleUrlAnnotationHandlerMapping maps the only declared url with Controller.
Accordingly, it should be declared with DefaultAnnotationHandlerMapping declared as follows.
SimpleUrlAnnotationHandlerMapping should have higher priority.

```
	<bean id="selectAnnotaionMapper"
		class="egovframework.rte.ptl.mvc.handler.SimpleUrlAnnotationHandlerMapping"
		p:order="1">
		<property name="interceptors">
			<list>
				<ref local="authenticInterceptor"/>
			</list>
		</property>
		<property name="urls">
			<set>
				<value>/*Employee.do</value>
				<value>/employeeList.do</value>
			</set>
		</property>
	</bean>

	<bean id="annotationMapper"

	class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping"
		p:order="2"/>

	<bean id="authenticInterceptor" class="com.easycompany.interceptor.AuthenticInterceptor"
/>
```

Reference

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