**Spring MVC Controller**

This is an example of Spring MVC Controllers. In Spring MVC, Controllers are used to provide access to the application behavior that is defined through a service interface. Controllers are the ones that interpret user input and transform it into a model that is represented to the user by the view.

Here we will create examples, making use of MultiActionController andParameterizableViewController, two controller implementations provided by Spring. In order to do so, we will create a simple application with a view and a Controller in each case, and we will add all configuration necessary to run the application.

All web.xml, spring-servlet.xml are same as other projects.

**MultiActionController**

MultiActionController is an implementation of Spring, that allows multiple request types to be handled by the same class. Subclasses of this class can handle several different types of request with methods of the form :

public (ModelAndView | Map | String | void) actionName(HttpServletRequest request, HttpServletResponse response);

A Map return value indicates a model that is supposed to be passed to a default view, whereas a String return value indicates the name of a view to be rendered without a specific model.

**public** **class** HelloWorldController **extends** MultiActionController {

**public** ModelAndView hello(HttpServletRequest request,

HttpServletResponse response) **throws** Exception {

ModelAndView model = **new** ModelAndView("helloWorld");

model.addObject("msg", "hello()");

**return** model;

}

**public** ModelAndView goodBye(HttpServletRequest request,

HttpServletResponse response) **throws** Exception {

ModelAndView model = **new** ModelAndView("helloWorld");

model.addObject("msg", "goodBye()");

**return** model;

}

}

In order to map the URL requests to the correct methods, MultiActionController makes use of the Resolvers. They are configured in the contoller bean definition in mvc-dispatcher-servlet.xml file, inside the methodNameResolver property. Here we shall see examples of three different resolver implementations provided by Spring.

### InternalPathMethodNameResolver

This is the default implementation used by the MultiActionController.

<bean

class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>

<property name=*"prefix"* value=*"/WEB-INF/views/"* />

<property name=*"suffix"* value=*".jsp"* />

</bean>

<bean

class=*"org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"* />

<bean class=*"in.spring4buddies.application.controller.HelloWorldController"*>

<property name=*"methodNameResolver"*>

<bean

class=*"org.springframework.web.servlet.mvc.multiaction.InternalPathMethodNameResolver"*>

</bean>

</property>

</bean>

<http://localhost:8080/sf-mvc3-old-multiactioncontroller/helloworld/hello.htm>

<http://localhost:8080/sf-mvc3-old-multiactioncontroller/helloworld/goodBye.htm>

InternalPathMethodNameResolver can make use of suffix and prefix attributes, that are applied to the initial URL request, so as to create the correct method name. For example, the below configuration will map requests like"/sf-mvc3-old-multiactioncontroller/helloworld/hello.htm"  
and "/sf-mvc3-old-multiactioncontroller/helloworld/hello.htm"

to sfHelloMessage() and sfGoodByeMessage() methods ofHelloWorldController respectively.

<bean class=*"in.spring4buddies.application.controller.HelloWorldController"*>

<property name=*"methodNameResolver"*>

<bean

class=*"org.springframework.web.servlet.mvc.multiaction.InternalPathMethodNameResolver"*>

<property name=*"prefix"* value=*"sf"* />

<property name=*"suffix"* value=*"Message"* />

</bean>

</property>

</bean>

### ParameterMethodNameResolver

This resolver has a property to configure, whose name is set to paramName and its value is set to action.

<bean class=*"in.spring4buddies.application.controller.HelloWorldController"*>

<property name=*"methodNameResolver"*>

<bean

class=*"org.springframework.web.servlet.mvc.multiaction.ParameterMethodNameResolver"*>

<property name=*"paramName"* value=*"action"* />

</bean>

</property>

</bean>

<http://localhost:8080/sf-mvc3-old-multiactioncontroller/helloworld/sfhelloMessage.htm?action=hello>

<http://localhost:8080/sf-mvc3-old-multiactioncontroller/helloworld/sfhelloMessage.htm?action=goodBye>

### PropertiesMethodNameResolver

This resolver maps URL requests to methods making use of properties as key-value pairs.

<bean class=*"in.spring4buddies.application.controller.HelloWorldController"*>

<property name=*"methodNameResolver"*>

<bean

class=*"org.springframework.web.servlet.mvc.multiaction.PropertiesMethodNameResolver"*>

<property name=*"mappings"*>

<props>

<prop key=*"/helloworld/hello.htm"*>hello</prop>

<prop key=*"/helloworld/goodBye.htm"*>goodBye</prop>

</props>

</property>

</bean>

</property>

</bean>

When running the application, the resolver will map "/sf-mvc3-old-multiactioncontroller/helloWorld/hello.htm" requests to the hello()method of HelloWorldController and "sf-mvc3-old-multiactioncontroller/helloWorld/goodBye.htm" requests to the goodBye() method ofHelloWorldController