**SPRING-MVC**

1. **Using default annotation based validation:**

Do not forget to add the following while using mvc validation:

1. **@Valid** annotation in handle method

2. **<mvc:annotation-driven />** in your config file

1. **Custom validation**

1. use @Component in validator class

2. use @Autowired for validator instance in controller

1. **Locale based messages**

To enable the internationalization in spring MVC. We need to register below 3 beans in the spring context.

* 1. **MessageSource**

<bean id="messageSource"

class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

<property name="basename" value="classpath:messages" />

<property name="defaultEncoding" value="UTF-8" />

</bean>

ReloadableResourceBundleMessageSource bean enables the internationalization (i18N).

basename property is used to provide the location of resource bundles.

value specifies that resource bundles are located at messages\_{locale}.properties.

defaultEncoding specifies the encoding used for the messages.

* 1. **LocaleResolver**

Resolves the locale based on its implementation.

It has following 3 implementations:

* + 1. **SessionLocaleResolver:**

resolves the locale based on the predefined attribute in the session.

* + 1. **CookieLocaleResolver :**

resolves the locale based on the predefined attribute in the cookie.

* + 1. **AcceptHeaderLocaleResolver :**

default implementation which resolves the locale by checking accept-language header in the HTTP request.

If we don’t define any locale resolvers , spring by default takes AcceptHeaderLocaleResolver.

Lets implement SessionLocaleResolver in our application.

Copy this code

<bean id="localeResolver" class="org.springframework.web.servlet.i18n.SessionLocaleResolver">

<property name="defaultLocale" value="en" />

</bean>

* 1. **LocaleChangeInterceptor**

This interceptor intercept the HTTP request and checks for the special parameter in the request.

The name of the parameter that it has to check in the request can be specified by the property called paramName, its value will be searched in the HTTP request by this interceptor.

So here , it searches for the parameter language in the HTTP request.Its value in the request is the locale value which it will add in the place of {locale} in the line messages\_ {locale} and searches for the corresponding messages property file.

<bean id="localeChangeInterceptor" class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">

<property name="paramName" value="language" />

</bean>

NOTE: <http://www.springframework.org/schema/beans/spring-beans-3.1.xsd> was needed in config.xml file to work.

1. **Using Localized messages in Java files** 
   1. Use message source in your class (any class )

@Autowired

**private** MessageSource messageSource;

* 1. Get Locale using following method:

//independent from the Servlet API and is based on a thread local in order to provide the current locale in any entity of your architecture. locale=LocaleContextHolder.*getLocale*();

4.3. Get message using following method:

messageSource.getMessage("validation.error.userName", null, "user name missing", locale))