LAB- Spring Mvc - Using Internationalization

In this lab you will be understanding how to create an internatinalized spring mvc application.

STEP -1

In this step, you will be working on **01internationalization-start** project

The code Provided to you is same as the code in **07mvc-tiles-solution**

If we want to support multiple languages, we need to create a ".properties" file for each language.

Firstly, you need to decide on a base name. Let the base name be "messages" .

So, if you want to support english and spanish, you need to create 2 files messages.properties (for english) and messages_es.properties (for spanish . "es" is the locale code for spanish).

Fortunately, these files have been created already for you. Open the file and observe the key value pairs.

1) Once you have the properties file, you need to configure a message source . Complete TODO-1 by configuring messageSource as shown Below :

```
<bean id="messageSource"
          class="org.springframework.context.support.ResourceBundleMessageSource">
          <property name="basename" value="messages" />
          <property name="cacheSeconds" value="60" />

     </bean>
```

2) If you want to show content as per the user's locale, you should use <spring:message code="key in .properties file" />

We want to show our topMenu in different languages as of now. So, Open topMenu.jsp under /WEB-INF/jsp/commons folder.

Replace "Browse Courses" with `<spring:message code="browsecourses"/>`

Observe that we have already used <spring:message /> tag where ever we want the dynamic language specific content.

3) Now, Deploy the application and give a request to /home.htm. You will observe that the top menu is displayed in English.

Change the browser's language to spanish and refresh the page. You should see that the top menu is displayed in spanish.

So, till now , the language is selected based on browser's language. Browser will send a header with Name "Accept" which contains the locale code.

If you dont configure a bean of type "LocaleResolver" , the default LocaleResolver is AcceptHeaderLocaleResolver which resolves the locale based on "Accept" header.

But we want the language to be selected based on a cookie.

4)Complete TODO-2 by configuring a CookieLocaleResolver as shown below :

```
  <bean id="localeResolver"
class="org.springframework.web.servlet.i18n.CookieLocaleResolver">
      <property name="cookieName" value="localecookie" />
      <property name="defaultLocale" value="en" />
      <property name="cookieMaxAge" value="3600" />
    </bean>
```

5) But who will set the cookie initially? and based on what parameter will be the cookie value set?

So, we need to define a Localchange interceptor Complete TODO-3 by configuring as below :

```
  <mvc:interceptors>
      <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor" />
    </mvc:interceptors>
```

This LocaleChangeInterceptor will look for a request parameter with name" **locale".** It will also look for a bean by type LocaleResolver. If there is CookieLocaleResolver, then this interceptor will add a Cookie.

Later, CookieLocaleResolver will resolve the cookie based on the cookie value.

6) If you observe topMenu.jsp, there are 2 links already configured with Name "English" and "Spanish" . Observe the links. If you click on Spanish link, request parameter " locale=es" will be sent. If you click on English Link, request parameter " locale=en" will be sent

Deploy the application, click on spanish link and english link and observe the change in the language.

Also, Observe the cookie in your browser. There will be a cookie with name "localecookie" sent from localhost

7) If you have time left, Similar to CookieLocaleResolver, experiment with SessionLocalResolver.

STEP 2

In this step, you will be working on **02internationalization-boot-start** project

The code given to you is same as the one given in **07mvc-tiles-using-boot-solution**

1) Press CTRL+SHIFT+T and open **MessageSourceAutoConfiguration .** Observe that the default basename is configured as **messages**

What if you want to change the base name?

you have to configure spring.messages.basename=mybasename in application.properties

For now, let us use basename as messages.
Observe that you have been provided with messages.properties and messages_es.properties in scr/main/resources.

So, even though you dont configure messageSource, Spring Boot will configure it.

2) Configure the following beans in **MvcConfig.java**

```
@Bean(name="localeResolver")
	public LocaleResolver LocaleResolver(){
		CookieLocaleResolver cookieLocaleResolver= new CookieLocaleResolver();
		cookieLocaleResolver.setCookieName("mylocalecookie");
		cookieLocaleResolver.setDefaultLocale(new Locale("en"));
		cookieLocaleResolver.setCookieMaxAge(3600);
		return cookieLocaleResolver;
	}

	@Bean
	public LocaleChangeInterceptor localeChangeInterceptor(){
		return new LocaleChangeInterceptor();
	}



	@Override
	public void addInterceptors(InterceptorRegistry registry) {
		registry.addInterceptor(localeChangeInterceptor());
	}
```

3) If you want to show content as per the user's locale, you should use <spring:message code="key in .properties file" />

We want to show our topMenu in different languages as of now. So, Open topMenu.jsp under /WEB-INF/jsp/commons folder.

Replace "Browse Courses" with `<spring:message code="browsecourses"/>`

Observe that we have already used <spring:message /> tag where ever we want the dynamic language specific content.

4) If you observe topMenu.jsp, there are 2 links already configured with Name "English" and "Spanish" . Observe the links. If you click on Spanish link, request parameter " locale=es" will be sent. If you click on English Link, request parameter " locale=en" will be sent

Deploy the application, click on spanish link and english link and observe the change in the language.

Congratulations !! You know how to internationalize the application.