Before moving forward please go through theory part of Spring MVC Interceptor from our notes, so that you can understand the concept of Spring MVC Interceptor well. For introductory theory part you can also refer comment section of file 'com.javaLive.controller.EmployeeController.java'

**What this project does??**

This is eighth project of Spring MVC project series.

**This is Full Spring MVC application with Java based configuration which demonstrates how to define Interceptors in Spring and how they work.**

This is web based maven project.

**Also note that here we are focusing on explanation of Spring Interceptor functionality only. Please go through project one to four of Spring MVC project series before this project so that you can understand the project structure and flow of request in the project well.**

' SpringMVCApplicationFlow': explains functionality of 'spring-servlet.xml' in Spring MVC context

' SpringMVCHelloWorld': explains how controller layer functions in Spring MVC

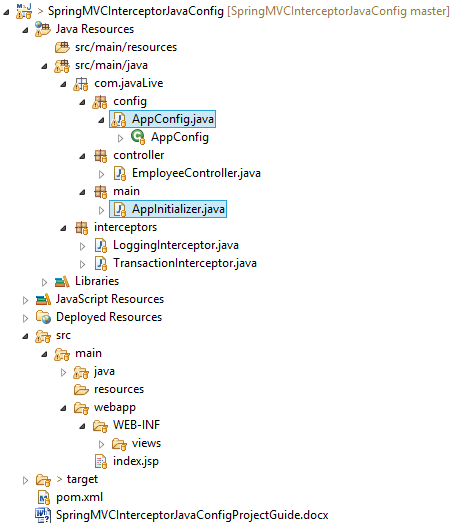
'SprinArithematicOperation': explains how controller layer and service layer functions in Spring MVC

'SpringMVCHibernateWithXMLFull' : explains full Spring+Hibernate database table CRUD operation **with full XML configuration** which includes all three layers of Spring MVC viz. Controller, Service and DAO.

**Note that this is Java configuration version of project 'SpringMVCInterceptorXMLConfig' so everything is similar except way of configuration as web.xml file is replaced by AppInitializer.java file and spring-servlet.xml file is replaced by AppConfig.java file (which contains configuration of Interceptors). So here only difference is way of configuring Spring Interceptors rest of the functioing is same as that of our previous project viz. '** **SpringMVCInterceptorXMLConfig'.**

**We do not have provided logging mechanism in this project.**

Project structure:

****

**In above project please go through files, packages and folders as follows:**

Packages config which contains configuration file for Spring which is Java counterpart of spring-servlet.xml.

Packages controller which contains controllers.

Packages main which contains AppInitializer class which is Java counterpart of web.xml file..

Packages interceptor which contains interceptor classes.

Folder WEB-INF/views which contains various JSP files.

File index.jsp which is welcome file of this project.

**Steps to create project:-**

1. This is web based maven project. Create webapp maven project. Please refer the file **'CreateWebBasedMavenProjectInEclipse.docx'** in this project to know more about creation of web based maven project.

2. Add require dependencies for spring as shown in pom.xml file.

3. Create required packages and add the files.

**Functioning of the project**

Here we are focusing on three parts only

1. How to define Interceptor?

For this purpose please go through theory of Spring MVC Interceptor in the notes and in comment section of file 'com.javaLive.controller.EmployeeController.java'**.**

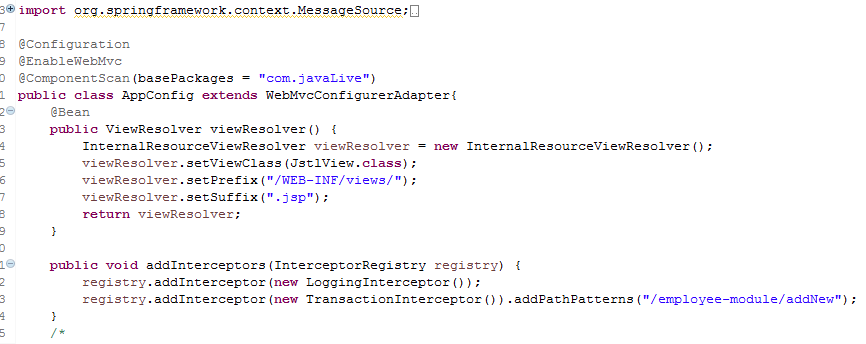
Now please go through the files in the package 'com.javaLive.interceptors' to study how to define Spring Interceptors.

Here in this package we have two interceptors viz. 'LoggingInterceptor' and 'TransactionInterceptor'. These interceptors do simple logging and displays various messages to console. But in real time application they plays vital role as:

Some Real-Time Usage of Interceptor :

* Maintaining Session / Session Management.
* Email Shoot & Database Check/logging.
* Authentication & Authorization Validations.
* Request Time Calculations. / Performance tunning. Etc.

2. How to configure Interceptor?



Above is the screen shot of part of ' com.javaLive.config.AppConfig.java' file which shows how to configure the Spring interceptors.

3. How Interceptor get executes during request flow?

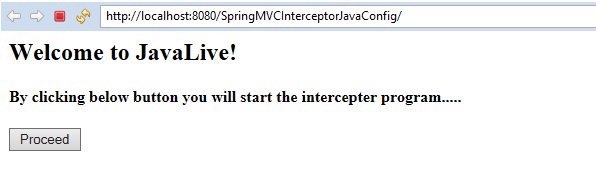
**Note a important difference between configuration of these two interceptors. In first configuration we haven't mentioned any path hence it is applicable to all the URIs of the application. Whereas second interceptor is specific to '/employee-module/addNew' URI only.**

If there are multiple spring interceptors configured, preHandle() method is executed in the order of configuration whereas postHandle() and afterCompletion() methods are invoked in the reverse order.

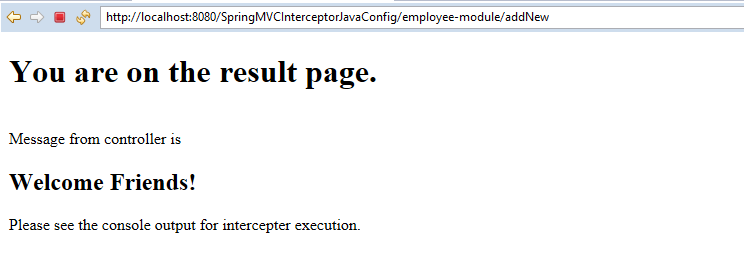
Flow of request in the project:

Right click on the project -> Run As -> Run On Server

It will shows following pageviz. index.jsp. Now click 'Proceed' button.

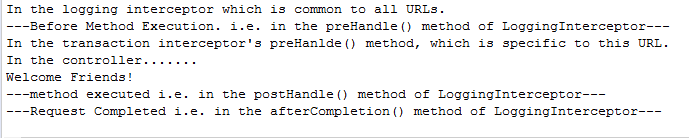


It shows result.jsp page as follows:



Now we have to look console output to study actual working of interceptors as they works behind the scene.

Please refer below screen shot of the console.



As you can see logging interceptor is first and is common interceptor for all URI, get executed first and TransactionInteceptor which is specific to given URI is get executed afterwards.

**\*\*Sequence of method execution of interceptor (Interview question)**

* All global interceptors gets executed first and then the mapped interceptor. Among them, the same order in which the interceptor are declared, the execution is also done.
* If true is returned, the execution chain continues and for false, the execution stops for that request with that interceptor.