## webmvc-config.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"* xmlns:context=*"http://www.springframework.org/schema/context"* xmlns:mvc=*"http://www.springframework.org/schema/mvc"* xmlns:p=*"http://www.springframework.org/schema/p"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd"*>  <!--  The controllers are autodetected POJOs labeled with the @Controller  annotation.  -->  <context:component-scan base-package=*"com.iptv.myehome"* use-default-filters=*"false"*>  <context:include-filter expression=*"org.springframework.stereotype.Controller"* type=*"annotation"* />  </context:component-scan>  <!--  Turns on support for mapping requests to Spring MVC @Controller  methods Also registers default Formatters and Validators for use  across all @Controllers  -->  <mvc:annotation-driven conversion-service=*"applicationConversionService"* />  <!--  Handles HTTP GET requests for /resources/\*\* by efficiently serving up  static resources  -->  <mvc:resources location=*"/, classpath:/META-INF/web-resources/"* mapping=*"/resources/\*\*"* />  <!--  Allows for mapping the DispatcherServlet to "/" by forwarding static  resource requests to the container's default Servlet  -->  <mvc:default-servlet-handler />  <!--  register "global" interceptor beans to apply to all registered  HandlerMappings  -->  <mvc:interceptors>  <bean class=*"org.springframework.web.servlet.theme.ThemeChangeInterceptor"* />  <bean class=*"org.springframework.web.servlet.i18n.LocaleChangeInterceptor"* p:paramName=*"lang"* />  </mvc:interceptors>  <!--  selects a static view for rendering without the need for an explicit controller  -->  <mvc:view-controller path=*"/"* view-name=*"index"* />  <mvc:view-controller path=*"/login"* />  <mvc:view-controller path=*"/tv"* view-name=*"/tv/index"* />  <mvc:view-controller path=*"/uncaughtException"* />  <mvc:view-controller path=*"/resourceNotFound"* />  <mvc:view-controller path=*"/dataAccessFailure"* />  <!--  Resolves localized messages\*.properties and application.properties  files in the application to allow for internationalization. The  messages\*.properties files translate Roo generated messages which are  part of the admin interface, the application.properties resource  bundle localizes all application specific messages such as entity  names and menu items.  -->  <bean class=*"org.springframework.context.support.ReloadableResourceBundleMessageSource"* id=*"messageSource"* p:basenames=*"WEB-INF/i18n/messages,WEB-INF/i18n/application"* p:fallbackToSystemLocale=*"false"* />  <!-- store preferred language configuration in a cookie -->  <bean class=*"org.springframework.web.servlet.i18n.CookieLocaleResolver"* id=*"localeResolver"* p:cookieName=*"locale"* />  <!--  resolves localized <theme\_name>.properties files in the classpath to  allow for theme support  -->  <bean class=*"org.springframework.ui.context.support.ResourceBundleThemeSource"* id=*"themeSource"* />  <!-- store preferred theme configuration in a cookie -->  <bean class=*"org.springframework.web.servlet.theme.CookieThemeResolver"* id=*"themeResolver"* p:cookieName=*"theme"* p:defaultThemeName=*"standard"* />  <!--  This bean resolves specific types of exceptions to corresponding  logical - view names for error views. The default behaviour of  DispatcherServlet - is to propagate all exceptions to the servlet  container: this will happen - here with all other types of exceptions.  -->  <bean class=*"org.springframework.web.servlet.handler.SimpleMappingExceptionResolver"* p:defaultErrorView=*"uncaughtException"*>  <property name=*"exceptionMappings"*>  <props>  <prop key=*".DataAccessException"*>dataAccessFailure</prop>  <prop key=*".NoSuchRequestHandlingMethodException"*>resourceNotFound</prop>  <prop key=*".TypeMismatchException"*>resourceNotFound</prop>  <prop key=*".MissingServletRequestParameterException"*>resourceNotFound</prop>  </props>  </property>  </bean>  <!--Installs application converters and formatters-->  <bean class=*"com.iptv.myehome.web.ApplicationConversionServiceFactoryBean"* id=*"applicationConversionService"* />  <!-- tiles视图  <bean class="org.springframework.web.servlet.view.UrlBasedViewResolver" id="tilesViewResolver">  <property name="viewClass" value="org.springframework.web.servlet.view.tiles2.TilesView" />  </bean>  <bean class="org.springframework.web.servlet.view.tiles2.TilesConfigurer" id="tilesConfigurer">  <property name="definitions">  <list>  <value>/WEB-INF/layouts/layouts.xml</value>  <value>/WEB-INF/jsp/\*\*/views.xml</value>  </list>  </property>  </bean>  -->  <!-- JSP视图 -->  <bean class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"* p:prefix=*"/WEB-INF/jsp/"* p:suffix=*".jsp"* />  <!-- 文件上传处理器 -->  <bean id=*"multipartResolver"* class=*"org.springframework.web.multipart.commons.CommonsMultipartResolver"* p:defaultEncoding=*"utf-8"* >  <property name=*"maxInMemorySize"*>  <value>4096</value>  </property>  </bean>  </beans> |

## web.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"ISO-8859-1"* standalone=*"no"*?>  <web-app xmlns=*"http://java.sun.com/xml/ns/j2ee"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* version=*"2.4"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd"*>  <display-name>myehome</display-name>  <description>Roo generated myehome application</description>  <!-- Enable escaping of form submission contents -->  <context-param>  <param-name>defaultHtmlEscape</param-name>  <param-value>true</param-value>  </context-param>  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath\*:META-INF/spring/applicationContext\*.xml</param-value>  </context-param>  <filter>  <filter-name>CharacterEncodingFilter</filter-name>  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  <init-param>  <param-name>encoding</param-name>  <param-value>UTF-8</param-value>  </init-param>  <init-param>  <param-name>forceEncoding</param-name>  <param-value>true</param-value>  </init-param>  </filter>  <filter>  <filter-name>HttpMethodFilter</filter-name>  <filter-class>org.springframework.web.filter.HiddenHttpMethodFilter</filter-class>  </filter>  <filter>  <filter-name>Spring OpenEntityManagerInViewFilter</filter-name>  <filter-class>org.springframework.orm.jpa.support.OpenEntityManagerInViewFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>CharacterEncodingFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <filter-mapping>  <filter-name>HttpMethodFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <filter-mapping>  <filter-name>Spring OpenEntityManagerInViewFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- Creates the Spring Container shared by all Servlets and Filters -->  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <!-- Handles Spring requests -->  <servlet>  <servlet-name>myehome</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/spring/webmvc-config.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>myehome</servlet-name>  <url-pattern>/</url-pattern>  </servlet-mapping>  <session-config>  <session-timeout>10</session-timeout>  </session-config>  <error-page>  <exception-type>java.lang.Exception</exception-type>  <location>/uncaughtException</location>  </error-page>  <error-page>  <error-code>404</error-code>  <location>/resourceNotFound</location>  </error-page>  </web-app> |

## persistence.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>  <persistence xmlns=*"http://java.sun.com/xml/ns/persistence"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* version=*"2.0"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence\_2\_0.xsd"*>  <persistence-unit name=*"persistenceUnit"* transaction-type=*"RESOURCE\_LOCAL"*>  <provider>org.hibernate.ejb.HibernatePersistence</provider>  <properties>  <property name=*"hibernate.dialect"* value=*"org.hibernate.dialect.MySQL5InnoDBDialect"*/>  <!--  value="create" to build a new database on each run;  value="update" to modify an existing database;  value="create-drop" means the same as "create" but also drops tables when Hibernate closes;  value="validate" makes no changes to the database  -->  <property name=*"hibernate.hbm2ddl.auto"* value=*"validate"*/>  <property name=*"hibernate.ejb.naming\_strategy"* value=*"org.hibernate.cfg.ImprovedNamingStrategy"*/>  <property name=*"hibernate.connection.charSet"* value=*"UTF-8"*/>  <property name=*"hibernate.show\_sql"* value=*"false"*/>  <!-- Uncomment the following two properties for JBoss only -->  <!-- property name="hibernate.validator.apply\_to\_ddl" value="false" /-->  <!-- property name="hibernate.validator.autoregister\_listeners" value="false" /-->  </properties>  </persistence-unit>  </persistence> |

## database.properties

|  |
| --- |
| #Updated at Tue Jun 07 13:50:04 CST 2011  #Tue Jun 07 13:50:04 CST 2011  #database.password=root  #database.url=jdbc\:mysql\://localhost\:3306/myehome\_xj  #database.username=root  #database.driverClassName=com.mysql.jdbc.Driver  database.password=myehome  database.url=jdbc\:mysql\://222.83.7.101\:3306/myehome\_xj  database.username=myehome  database.driverClassName=com.mysql.jdbc.Driver |

## applicationContext.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"* xmlns:aop=*"http://www.springframework.org/schema/aop"* xmlns:context=*"http://www.springframework.org/schema/context"* xmlns:jee=*"http://www.springframework.org/schema/jee"* xmlns:tx=*"http://www.springframework.org/schema/tx"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.0.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd"*>  <!--  This will automatically locate any and all property files you have  within your classpath, provided they fall under the META-INF/spring  directory. The located property files are parsed and their values can  then be used within application context files in the form of  ${propertyKey}.  -->  <context:property-placeholder location=*"classpath\*:META-INF/spring/\*.properties"*/>  <!--  Turn on AspectJ @Configurable support. As a result, any time you  instantiate an object, Spring will attempt to perform dependency  injection on that object. This occurs for instantiation via the "new"  keyword, as well as via reflection. This is possible because AspectJ  is used to "weave" Roo-based applications at compile time. In effect  this feature allows dependency injection of any object at all in your  system, which is a very useful feature (without @Configurable you'd  only be able to dependency inject objects acquired from Spring or  subsequently presented to a specific Spring dependency injection  method). Roo applications use this useful feature in a number of  areas, such as @PersistenceContext injection into entities.  -->  <context:spring-configured/>  <!--  This declaration will cause Spring to locate every @Component,  @Repository and @Service in your application. In practical terms this  allows you to write a POJO and then simply annotate the new POJO as an  @Service and Spring will automatically detect, instantiate and  dependency inject your service at startup time. Importantly, you can  then also have your new service injected into any other class that  requires it simply by declaring a field for your service inside the  relying class and Spring will inject it. Note that two exclude filters  are declared. The first ensures that Spring doesn't spend time  introspecting Roo-specific ITD aspects. The second ensures Roo doesn't  instantiate your @Controller classes, as these should be instantiated  by a web tier application context. Refer to web.xml for more details  about the web tier application context setup services.    Furthermore, this turns on @Autowired, @PostConstruct etc support. These  annotations allow you to use common Spring and Java Enterprise Edition  annotations in your classes without needing to do any special configuration.  The most commonly used annotation is @Autowired, which instructs Spring to  dependency inject an object into your class.  -->  <context:component-scan base-package=*"com.iptv.myehome"*>  <context:exclude-filter expression=*".\*\_Roo\_.\*"* type=*"regex"*/>  <context:exclude-filter expression=*"org.springframework.stereotype.Controller"* type=*"annotation"*/>  </context:component-scan>  <bean class=*"org.apache.commons.dbcp.BasicDataSource"* destroy-method=*"close"* id=*"dataSource"*>  <property name=*"driverClassName"* value=*"${database.driverClassName}"*/>  <property name=*"url"* value=*"${database.url}"*/>  <property name=*"username"* value=*"${database.username}"*/>  <property name=*"password"* value=*"${database.password}"*/>  <property name=*"validationQuery"* value=*"SELECT 1 FROM DUAL"*/>  <property name=*"testOnBorrow"* value=*"true"*/>  </bean>  <bean class=*"org.springframework.orm.jpa.JpaTransactionManager"* id=*"transactionManager"*>  <property name=*"entityManagerFactory"* ref=*"entityManagerFactory"*/>  </bean>  <tx:annotation-driven mode=*"aspectj"* transaction-manager=*"transactionManager"*/>  <bean class=*"org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean"* id=*"entityManagerFactory"*>  <property name=*"dataSource"* ref=*"dataSource"*/>  </bean>  </beans> |