CIDC 系統search查詢

<context-param>

<param-name>contextConfigLocation</param-name <param-value>/WEB-INF/applicationContext-\*.xml,/WEB-INF/security.xml</param-value>

</context-param>

<servlet>

<servlet-name>action</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<load-on-startup>2</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>action</servlet-name>

<url-pattern>\*.html</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>action</servlet-name>

<url-pattern>\*.service</url-pattern>

</servlet-mapping>

首先定義了spring MVC的啟動配置，在action-servlet.xml文件中定義了視圖解析器以及控制器

，如下：

<!-- View Resolver for JSPs -->

<bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="requestContextAttribute" value="rc"/>

<property name="viewClass" value="org.springframework.web.servlet.view.JstlView"/>

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

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

</bean>

控制器

<bean id="urlMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">

<property name="mappings">

<props>

<prop key="/editProfile.html">userFormController</prop>

<prop key="/mainMenu.html">filenameController</prop>

<prop key="/editUser.html">userFormController</prop>

<prop key="/selectFile.html">fileUploadController</prop>

<prop key="/uploadFile.html">fileUploadController</prop>

<prop key="/passwordHint.html">passwordHintController</prop>

<prop key="/signup.html">signupController</prop>

<prop key="/news.service">httpNewsManager</prop>

<prop key="/newsImage.service">httpNewsImageManager</prop>

<prop key="/newsImageHelper.service">httpNewsImageHelper</prop>

<prop key="/website.service">httpWebsiteManager</prop>

<prop key="/keyword.service">httpKeywordManager</prop>

<prop key="/website.service">httpWebsiteManager</prop>

<prop key="/price.service">httpPriceManager</prop>

<prop key="/exchangeRate.service">httpExchangeRateManager</prop>

<prop key="/interestRate.service">httpInterestRateManager</prop>

<prop key="/search.html">searchController</prop>

<prop key="/reindex.html">indexController</prop>

<prop key="/indexManager.html">indexManagerController</prop>

<prop key="/searchWebsite.html">websiteSearchController</prop>

<prop key="/reindexWebsite.html">websiteIndexController</prop>

<!--Keyword-URL-START-->

<prop key="/keywords.html">keywordController</prop>

<prop key="/editKeyword.html">keywordFormController</prop>

<!--Keyword-URL-END-->

<!--CatalogType-URL-START-->

<prop key="/catalogTypes.html">catalogTypeController</prop>

<prop key="/editCatalogType.html">catalogTypeFormController</prop>

<!--CatalogType-URL-END-->

<!--Catalog-URL-START-->

<prop key="/catalogs.html">catalogController</prop>

<prop key="/editCatalog.html">catalogFormController</prop>

<!--Catalog-URL-END-->

<!--News-URL-START-->

<prop key="/newss.html">newsController</prop>

<prop key="/editNews.html">newsFormController</prop>

<!--News-URL-END-->

<!--Website-URL-START-->

<prop key="/websites.html">websiteController</prop>

<prop key="/editWebsite.html">websiteFormController</prop>

<!--Website-URL-END-->

<!--NewsImage-URL-START-->

<prop key="/newsImages.html">newsImageController</prop>

<prop key="/editNewsImage.html">newsImageFormController</prop>

<!--NewsImage-URL-END-->

<!--Material-URL-START-->

<prop key="/materials.html">materialController</prop>

<prop key="/editMaterial.html">materialFormController</prop>

<!--Material-URL-END-->

<!--Price-URL-START-->

<prop key="/prices.html">priceController</prop>

<prop key="/editPrice.html">priceFormController</prop>

<!--Price-URL-END-->

<!--ExchangeRate-URL-START-->

<prop key="/exchangeRates.html">exchangeRateController</prop>

<prop key="/editExchangeRate.html">exchangeRateFormController</prop>

<!--ExchangeRate-URL-END-->

<!--InterestRate-URL-START-->

<prop key="/interestRates.html">interestRateController</prop>

<prop key="/editInterestRate.html">interestRateFormController</prop>

<!--InterestRate-URL-END-->

<!--UserActivityLog-URL-START-->

<prop key="/userActivityLogs.html">userActivityLogController</prop>

<prop key="/editUserActivityLog.html">userActivityLogFormController</prop>

<!--UserActivityLog-URL-END-->

<!--Quoter-URL-START-->

<prop key="/quoters.html">quoterController</prop>

<prop key="/editQuoter.html">quoterFormController</prop>

<!--Quoter-URL-END-->

<!--JudgeCase-URL-START-->

<prop key="/judgeCases.html">judgeCaseController</prop>

<prop key="/editJudgeCase.html">judgeCaseFormController</prop>

<!--JudgeCase-URL-END-->

<!-- Add additional URL mappings here -->

</props>

</property>

</bean>

<bean id="adminUrlMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">

<property name="interceptors">

<list>

<ref bean="adminInterceptor"/>

</list>

</property>

<property name="mappings">

<props>

<prop key="/activeUsers.html">filenameController</prop>

<prop key="/users.html">userController</prop>

<prop key="/flushCache.html">filenameController</prop>

<prop key="/reload.html">reloadController</prop>

</props>

</property>

</bean>

現在index頁面中加載search頁面，當點擊查詢時，提交表單請求到search.html,在UrlMapping中加請求交給searchController處理

<bean id="searchController"

class="com.foxconn.cic.webapp.action.NewsSearchController">

<property name="compass">

<ref bean="compass" />

</property>

<property name="searchView">

<value>search</value>

</property>

<property name="searchResultsView">

<value>search</value>

</property>

<property name="newsFacade" ref="newsFacade">

</property>

</bean>

searchController這個bean是NewsSearchController的實類，該類繼承了compass的類AbstractCompassCommandController(是compass中的類，它是AbstractCommandController的子類，所以是一種特殊的控制器，用來獲取compass，compassTemplate，以及設置compass)，NewsSearchController有幾個屬性，newsFacade是它的一個私有屬性，是NewsFacade的實類，

NewsFacade是接口，該接口的唯一實現類是NewsFacadeImpl。

當請求交給serachCOntroller是，會自動執行類中的handle方法。Handle方法通過compass機制，講查詢到的結果集返回到search頁面，然後顯示。

現在來看compass是如何執行查詢的。

在開始執行的時候，先將查詢的條件封裝在AdvancedSearchCommand對象searchCommand中，然後調用NewsFacade對象newsFacade的search方法，講searchCommand作為參數，進行查詢。將查詢的結果按照映射的方式保存後，返回在search頁面，予以顯示。

下麵來看，search方法是怎麼進行查詢，NewsFacade只是一個接口，實現類是NewsFacadeImpl。該類只包含一個NewsManager的屬性，通過該屬性來執行相應的查詢操作。

NewsManager是一個接口，其中定義了一系列的方法：

getNewss(News news)獲得新聞信息

getNewss(final String beginId,final String endId)獲得新聞信息

getNewIds() 獲得新聞的ID序列

getNews(final String id)通過Id來獲得該條新聞

saveNews(News news)保存新聞

removeNews(final String id)移除新聞

getNews(String title,String url)通過地址和標題來過的新聞

CompassSearchResults search(CompassSearchCommand searchCommand)定義查詢方式

該接口的實現類是NewsManagerImpl，NewsManagerImpl不僅實現了NewsManager的接口，同時也繼承了BaseManager的父類。

NewsManagerImpl自身定義了三個屬性。

**private** NewsDao dao;

**private** ImageTagHelper imageTagHelper;

**private** CompassSearchHelper searchHelper;

在這個類中，可以找到search的最終實現

**public** CompassSearchResults search(CompassSearchCommand searchCommand) {

CompassSearchResults result = searchHelper.search(searchCommand);

**return** result;

}

可以看出，search其實就是compass中CompassSearchHelper中自帶的方法。

NewsDao對象負責對NewsManager中抽象方法的實現，如：獲得新聞getNewss()，獲得新聞geteNEws()等等。

NewsDao其實也是接口，它的實現類是NewsDaoHibernate。對數據的操作用到了hibernate的Criteria.

接下來看NewsManagerImpl的另一個屬性， ImageTagHelper imageTagHelper.該屬性用來對新聞中的圖片進行操作。當遇到新聞中有圖片的時候，需要講圖片保存的另外的數據表中，而內容中用變量來代替，顯示的時候，講變量替換為圖片連接，然後將圖片顯示出來。