

**WSC FRAMEWORK 개발자 가이드**



2015 June 11

C-NODE

|  |  |  |  |
| --- | --- | --- | --- |
| **Current Version** | | 0.4 | |
| **Author** | | C-NODE | |
| **Date Created** | | 2014.10.20 | |
| **Approved By** | |  | |
| **Approval Date** | |  | |
|  |  |  |  |
|  |  |  |  |
| **Revision History** | |  |  |
|  |  |  |  |
| **Version no.** | **Date Updated** | **Revision Author** | **Summary of Major Changes Made** |
| 0.1 | 2014.10.20 | L.S.H | Initial document created. |
| 0.2 | 2015.05.27 | L.S.H | Framework improvement |
| 0.3 | 2015.06.11 | L.S.H | Framework improvement |
| 0.4 | 2016.12.01 | K.Y.J | iReport guide added |
|  |  |  |  |

목 차

[프레임워크 개요 5](#_Toc468373427)

[1. 프레임워크 특징 5](#_Toc468373428)

[2. 프레임워크 개발 환경 5](#_Toc468373429)

[3. 프레임워크 라이브러리 구성요소 5](#_Toc468373430)

[4. 프레임워크 구조 7](#_Toc468373431)

[5. 프레임워크 설정 파일 8](#_Toc468373432)

[6. 프레임워크 명명 규칙 15](#_Toc468373433)

[7. 자바소스 파일목록 20](#_Toc468373434)

[8. 스크립트 공통파일 23](#_Toc468373435)

[기능별 설명 24](#_Toc468373436)

[1. Code Cache 설정 24](#_Toc468373437)

[2. 시스템 ID 설정 24](#_Toc468373438)

[3. 인증 및 권한 설정 24](#_Toc468373439)

[4. 관리자의 자동 로그인 기능 25](#_Toc468373440)

[5. 사용자 그룹 26](#_Toc468373441)

[6. 화면 및 메뉴 권한 설정 26](#_Toc468373442)

[7. Pagination 기능 27](#_Toc468373443)

[8. Web Editor 기능 30](#_Toc468373444)

[9. 첨부파일 Control 기능 30](#_Toc468373445)

[10. Combo Box & Combo Grid 기능 32](#_Toc468373446)

[11. Excel Download 기능 35](#_Toc468373447)

[12. Excel Loader 기능 36](#_Toc468373448)

[13. Excel Loader 맵핑 기능 39](#_Toc468373449)

[14. EasyUI Grid 기능 41](#_Toc468373450)

[15. Context Menu 기능 42](#_Toc468373451)

[16. Hot Menu 기능 42](#_Toc468373452)

[17. 일정관리 기능 43](#_Toc468373453)

[18. 전자메일 기능 44](#_Toc468373454)

[19. iReport 연동 기능 46](#_Toc468373455)

[20. iReport Jaspersoft연동 48](#_Toc468373456)

[21. EasyUI Theme 변경 56](#_Toc468373457)

[22. 공지사항 (페이지 이동형 게시판) 57](#_Toc468373458)

[23. 팝업관리 57](#_Toc468373459)

[24. 개발 참고 샘플 58](#_Toc468373460)

[프로젝트 개발환경 설정방법 60](#_Toc468373461)

[1. 개발환경 설치 60](#_Toc468373462)

[2. 이클립스 내에 SVN 설정 및 소스 다운로드 61](#_Toc468373463)

[3. 이클립스 내의 WAS 설정 61](#_Toc468373464)

[4. 프로젝트 환경 설정 61](#_Toc468373465)

[5. WAS 실행 및 첫 페이지 구동 결과 확인 62](#_Toc468373466)

[샘플게시판 개발 가이드 63](#_Toc468373467)

[1. 게시판 그룹 추가 64](#_Toc468373468)

[2. 첨부파일 정보 추가 65](#_Toc468373469)

[3. 게시판 클래스 생성 65](#_Toc468373470)

[4. JSP and JavaScript 생성 67](#_Toc468373471)

[5. WAS 실행 67](#_Toc468373472)

[6. 메뉴 생성 및 권한 설정 67](#_Toc468373473)

[7. 결과 확인 70](#_Toc468373474)

[이후 개발 과제 72](#_Toc468373475)

[1. 현재까지 구현된 기능 72](#_Toc468373476)

[2. 아직 개발되지 않은 기능 74](#_Toc468373477)

[3. 보완 필요 사항 74](#_Toc468373478)

[프레임워크 개선 사항 75](#_Toc468373479)

[1. 1차 개선 사항 (2015년04월 – 2015년06월) 75](#_Toc468373480)

# 프레임워크 개요

## 프레임워크 특징

* **Spring 3.2.8 버전을 활용한 프레임워크 구성**
* **SLF4J and Log4j 기반 Logging**
* **Mybatis 를 사용한 Data Access Control**
* **Ehcache 를 활용한 공통코드의 Caching 처리**
* **EasyUI 기반의 웹 화면 구현**
* **iReport를 사용한 Reporting 및 Printing 구현**
* **Database에 기반한 사용자의 화면 접근 권한 설정**
* **FullCalendar jQuery Plugin 을 사용한 일정관리 기능**
* **ScEditor jQuery Plugin 을 사용하여 웹 에디터 구현**
* **Upload File jQuery Plugin을 사용하여 다중 파일 업로드 및 파일의 Drag and Drop 구현**

## 프레임워크 개발 환경

* **Windows 8 64bit**
* **JDK 1.6.0\_30 64bit**
* **Apache Tomcat 7.0.27**
* **Oracle 11.2.0.0.0**
* **Eclipse IDE for Java Developers - Luna Package**

## 프레임워크 라이브러리 구성요소

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Section** | **Library** | **Version** | **URL** |
| 프레임워크 | Core | springframework | 3.2.8 | spring.io |
| 프레임워크 | Logging | slf4j | 1.6.6 | www.slf4j.org |
| 프레임워크 | Caching | ehcache | 2.8.3 | ehcache.org |
| 프레임워크 | Database | mybatis | 3.2.3 | mybatis.github.io |
| 프레임워크 | View | spring MVC | 3.2.8 | spring.io |
| 프레임워크 | JDBC | oracle | 10.2.0.4.0 |  |
| 프레임워크 | Core | servlet-api | 3.0.1 |  |
| 프레임워크 | Core | jsp-api | 2.2 |  |
| 프레임워크 | Core | el | 2.2 |  |
| 프레임워크 | Core | jstl | 1.2 |  |
| 프레임워크 | Excel | jxls | 1.0.5 | jxls.sourceforge.net |
| 프레임워크 | View | json-simple | 1.1 | code.google.com/p/json-simple |
| 프레임워크 | View | jackson | 1.9.13 | jackson.codehaus.org |
| 화면UI | WEB | jquery-easyui | 1.4.0 | jeasyui.com |
| 화면UI | WEB | jquery-sceditor | 1.4.5 | sceditor.com |
| 화면UI | WEB | DaumEditor | 7.5.9 | github.com/daumcorp/DaumEditor |
| 화면UI | WEB | jquery-ckeditor | 4.4.7 | ckeditor.com |
| 화면UI | WEB | jquery-uploadfile | 3.1 | hayageek.com/docs/jquery-upload-file.php |
| 화면UI | WEB | jquery-json | 2.3 | www.json.org |
| 화면UI | WEB | jquery-fullcalendar | 2.1.1 | fullcalendar.io |
| 프레임워크 | WEB | iReport | 5.6.0 | sourceforge.net/projects/ireport/ |

## 프레임워크 구조

|  |  |
| --- | --- |
|  | 1. 자바소스 폴더:   src/main/java   1. 자바리소스 폴더:   src/main/resources   1. Mybatis 맵핑 SQL 폴더   src/main/resources/mappers   1. 메시지 리소스 폴더   src/main/resources/messages   1. 엑셀 Loader 폴더   src/main/resources/xlsloaders   1. 라이브러리 폴더:   src/main/webapp/WEB-INF/lib   1. 웹 리소스 폴더:   src/main/webapp/resources   1. 웹JSP 폴더:   src/main/webapp/WEB-INF/views   1. 스프링설정 폴더:   src/main/webapp/WEB-INF/spring   1. 엑셀 템플릿 폴더:   src/main/webapp/WEB-INF/views   1. iReport 소스 폴더:   src/main/webapp/WEB-INF/report   1. 문서 폴더: doc 2. 파일업로드 폴더: upload |

## 프레임워크 설정 파일

|  |  |  |
| --- | --- | --- |
| 설정파일 경로 | 설정파일명 | 설명 |
| src/main/resources/ | app.properties | Database 연결 정보  Security 설정 정보  Default 설정 정보  물리적 경로 및 도메인 정보 |
| src/main/resources/ | mybatis.xml | Mybatis 공통 설정 정보 |
| src/main/resources/ | log4j.xml | Logging 설정 정보 |
| src/main/resources/ | ehcache.xml | Caching 설정 정보 |
| src/main/webapp/WEB-INF/ | web.xml | WAS 의 Context 설정 정보 |
| src/main/webapp/WEB-INF/spring/ | wsc-context.xml | Spring 기본 설정 정보 |
| src/main/webapp/WEB-INF/ spring/ | wsc-mybatis.xml | Spring Database 관련 설정 정보 |
| src/main/webapp/WEB-INF/ spring/ | wsc-jasper.xml | Spring iReport 연동 설정 정보 |

1. **app.properties**

|  |  |
| --- | --- |
| 항목 | 설명 |
| jdbc.driver=oracle.jdbc.driver.OracleDriver  jdbc.url=jdbc:oracle:thin:@{IP}:{PORT}:{DB명}  jdbc.user={계정ID}  jdbc.pwd={계정비밀번호} | **Database 연결 계정정보** |
| security.form.username=j\_userid  security.form.password=j\_password  security.form.targeturl=target\_url  security.form.remember=j\_remember  security.form.language=j\_language  security.form.system=j\_system  security.form.secure=j\_secure | **Security 인증 관련 정보**  로그인 폼에서 사용할 파라메터의 명칭 정보를 정의한다.  자세한 내용은 SecurityIntercepter 클래스를 참조한다. |
| security.page.submit=/security\_check.do  security.page.login=/login.do  security.page.logout=/logout.do  security.page.default=/index.do  security.page.error=/denied.do | **Security URL 관련 정보**  로그인, 로그아웃, 메인, 접근에러에 관한 URL 을 정의한다.  자세한 내용은 SecurityIntercepter 클래스를 참조한다. |
| security.guest.id=guest | **Security 비인증 사용자 기본 ID** |
| default.system.id=WSC  default.page.size=50  loggable.userLog=false  usable.tabPanel=true | 기본 시스템 ID  기본 페이징 출력수  사용자DB로깅여부 (true인 경우 SYS\_ULOG에 저장)  탭패널사용여부 (true인 경우 탭패널로 화면 오픈) |
| file.upload.temp={프로젝트경로}/upload/temp  file.upload.real={프로젝트경로}/upload/real | 첨부파일 업로드 관련 물리적 경로 |
| excel.template.path=  {프로젝트경로}/src/main/webapp/WEB-INF/templates  excel.template.temp={프로젝트경로}/upload/excel | 엑셀 다운로드 관련 템플릿 경로 및 임시파일 저장 경로 |
| excel.loader=/xlsloaders/ | 엑셀 로더 사용시 설정파일 경로 |
| domain.user=user.local.com:8080 | 관리자의 사용자별 자동로그인 사용시 별도의 가상호스트 도메인 정보를 정의한다. |

1. **mybatis.xml**

|  |  |
| --- | --- |
| 항목 | 설명 |
| <settings>  <setting name=*"cacheEnabled"* value=*"true"*/>  <setting name=*"lazyLoadingEnabled"* value=*"false"*/>  <setting name=*"multipleResultSetsEnabled"* value=*"true"*/>  <setting name=*"useColumnLabel"* value=*"true"*/>  <setting name=*"useGeneratedKeys"* value=*"false"*/>  <setting name=*"defaultExecutorType"* value=*"SIMPLE"*/>  <setting name=*"defaultStatementTimeout"* value=*"25000"*/>  <setting name=*"logImpl"* value=*"LOG4J"*/>  </settings> | **Mybatis 의 기본 설정정보**  자세한 내용은 Mybatis 매뉴얼을 참조한다. |
| <typeAliases>  <typeAlias alias=*"params"* type=*"com.wsc.framework.model.ParamsMap"* />  <typeAlias alias=*"record"* type=*"com.wsc.framework.model.RecordMap"* />  <typeAlias alias=*"code"* type=*"com.wsc.common.model.Code"* />  <typeAlias alias=*"user"* type=*"com.wsc.common.model.User"* />  <typeAlias alias=*"menu"* type=*"com.wsc.common.model.Menu"* />  <typeAlias alias=*"group"* type=*"com.wsc.common.model.Group"* />  <typeAlias alias=*"prog"* type=*"com.wsc.common.model.Program"* />  <typeAlias alias=*"file"* type=*"com.wsc.common.model.FileInfo"* />  </typeAliases> | **공통으로 사용하는 모델의 Alias 정보** |

1. **log4j.xml**

* 현재 로그파일은 톰캣의 logs 밑으로 설정되어 있으며, 만약 해당 로그 파일의 위치를 변경하고자 할 때 DailyRollingFileAppender 의 경로를 변경하도록 한다.

|  |  |
| --- | --- |
| 항목 | 설명 |
| <appender name=*"console"* class=*"org.apache.log4j.ConsoleAppender"*></appender>  <appender name=*"wscapp"* class=*"org.apache.log4j.DailyRollingFileAppender"*></appender>  <appender name=*"wscsql"* class=*"org.apache.log4j.DailyRollingFileAppender"*></appender> | **Logging 시 해당 Target 정보** |
| <logger name=*"org.springframework.core"*></logger>  <logger name=*"org.springframework.beans"*></logger>  <logger name=*"org.springframework.context"*></logger>  <logger name=*"org.springframework.web"*></logger>  <logger name=*"com.wsc"*></logger>  <logger name=*"org.apache.jasper.compiler.TldLocationsCache"*></logger>  <root></root> | **어플리케이션별 Logging 설정** |

1. **ehcache.xml**

|  |  |
| --- | --- |
| 항목 | 설명 |
| <cache name=*"codeCache"*  maxElementsInMemory=*"10000"*  eternal=*"false"*  timeToIdleSeconds=*"0"*  timeToLiveSeconds=*"600"*  overflowToDisk=*"false"*  diskPersistent=*"false"*  diskExpiryThreadIntervalSeconds=*"120"*  memoryStoreEvictionPolicy=*"LRU"*>  </cache> | **공통코드 검색 Caching 에 관련한 설정 정보** |

1. **web.xml**

|  |  |
| --- | --- |
| 항목 | 설명 |
| <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/spring/wsc-context.xml</param-value>  </context-param> | **Spring Context 설정파일 정의** |
| <listener><listener-class>org.springframework.web.context.ContextLoaderListener</listener-class></listener>  <listener><listener-class>org.springframework.web.context.request.RequestContextListener</listener-class></listener> | **리스너 설정** |
| <filter>  <filter-name>encodingFilter</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>  </filter>  <filter-mapping>  <filter-name>encodingFilter</filter-name>  <url-pattern>\*.do</url-pattern>  </filter-mapping>  <filter-mapping>  <filter-name>encodingFilter</filter-name>  <url-pattern>\*.json</url-pattern>  </filter-mapping> | **한글 인코딩 설정** |
| <servlet>  <servlet-name>wscServlet</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/spring/wsc-context.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>default</servlet-name>  <url-pattern>/resources/\*</url-pattern>  </servlet-mapping>  <servlet-mapping>  <servlet-name>default</servlet-name>  <url-pattern>\*.ico</url-pattern>  </servlet-mapping>  <servlet-mapping>  <servlet-name>wscServlet</servlet-name>  <url-pattern>/</url-pattern>  </servlet-mapping> | **기본 Servlet 설정** |
| <error-page>  <error-code>403</error-code>  <location>/WEB-INF/views/error/denied.jsp</location>  </error-page>  <error-page>  <error-code>404</error-code>  <location>/WEB-INF/views/error/error.jsp</location>  </error-page>  <error-page>  <error-code>500</error-code>  <location>/WEB-INF/views/error/error.jsp</location>  </error-page> | **에러페이지 설정** |
| <welcome-file-list>  <welcome-file>index.do</welcome-file>  </welcome-file-list> | **디폴트 페이지 설정** |

1. **wsc-context.xml**

* 현재 로그파일은 톰캣의 logs 밑으로 설정되어 있으며, 만약 해당 로그 파일의 위치를 변경하고자 할 때 DailyRollingFileAppender 의 경로를 변경하도록 한다.

|  |  |
| --- | --- |
| 항목 | 설명 |
| <mvc:default-servlet-handler /> | **기본 Servlet 을 Spring의 DispatcherServlet 으로 정의** |
| <mvc:resources mapping=*"/resources/\*\*"* location=*"/resources/"* />  <mvc:resources mapping=*"/images/\*\*"* location=*"/images/"* />  <mvc:resources mapping=*"/css/\*\*"* location=*"/css/"* /> | **Spring 의 DispatcherServlet 에서 인식할 정적인 자원 정의** |
| <mvc:annotation-driven /> | **Spring MVC Controller 를 @Controller 라는 Annotation 이 선언된 클래스를 자동으로 바인딩 한다.** |
| <context:component-scan base-package=*"com.wsc"*/> | **Spring 의 Component Scan시 기준이 되는 Package 정보** |
| <util:properties id=*"app"* location=*"classpath:app.properties"*/> | **사용할 Properties 파일 정의** |
| <bean id=*"wscCacheManager"* class=*"org.springframework.cache.ehcache.EhCacheCacheManager"*>  <property name=*"cacheManager"*>  <bean class=*"org.springframework.cache.ehcache.EhCacheManagerFactoryBean"*>  <property name=*"configLocation"* value=*"classpath:ehcache.xml"*/>  <property name=*"shared"* value=*"true"*/>  </bean>  </property>  </bean>  <cache:annotation-driven cache-manager=*"wscCacheManager"* /> | **코드 Caching 에 사용할 Cache Manager 정의** |
| <bean id=*"sessionFactory"* class=*"org.springframework.beans.factory.config.ObjectFactoryCreatingFactoryBean"*>  <property name=*"targetBeanName"* value=*"sessionComponent"* />  </bean> | **공통으로 사용할 Session 객체 정의** |
| <bean class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>  <property name=*"prefix"* value=*"/WEB-INF/views/"* />  <property name=*"suffix"* value=*".jsp"* />  <property name=*"order"* value=*"99"* />  </bean>  <bean class=*"org.springframework.web.servlet.view.BeanNameViewResolver"*>  <property name=*"order"* value=*"0"* />  </bean> | **기본적으로 사용할 View Resolver 정의** |
| <bean class=*"org.springframework.web.servlet.view.XmlViewResolver"*>  <property name=*"location"* value=*"/WEB-INF/spring/wsc-jasper.xml"*/>  <property name=*"order"* value=*"1"*/>  </bean>  <import resource=*"wsc-jasper.xml"*/> | **iReport 연동을 위한 View Resolver 와 설정파일 Import** |
| <bean id=*"multipartResolver"* class=*"org.springframework.web.multipart.commons.CommonsMultipartResolver"*>  <property name=*"maxUploadSize"* value=*"104857600"* />  <property name=*"maxInMemorySize"* value=*"10485760"* /> <property name=*"uploadTempDir"* ref=*"uploadDirResource"* />  </bean>  <bean id=*"uploadDirResource"* class=*"org.springframework.core.io.FileSystemResource"*>  <constructor-arg><value>#{app['file.upload.temp']}</value> </constructor-arg>  </bean> | **첨부파일 업로드 관련 View Resolver 와 첨부파일 물리적 경로 설정** |
| <bean class=*"org.springframework.web.servlet.handler.SimpleMappingExceptionResolver"*>  <property name=*"warnLogCategory"* value=*"apperror"* />  <property name=*"defaultErrorView"* value=*"error/system"* />  <property name=*"exceptionMappings"*>  <props>  <prop key=*"com.wsc.framework.exception.SessionException"* >error/session</prop>  <prop key=*"com.wsc.framework.exception.AuthorityException"*>error/authority</prop>  <prop key=*"com.wsc.framework.exception.ServiceException"* >error/service</prop>  <prop key=*"com.wsc.framework.exception.SystemException"* >error/system</prop>  <prop key=*"java.lang.Exception"* >error/system</prop>  </props>  </property>  </bean> | **Exception Handling 에 필요한 Reolver 정의** |
| <bean id=*"jsonView"* class=*"org.springframework.web.servlet.view.json.MappingJacksonJsonView"* />  <bean id=*"jxlsView"* class=*"com.wsc.framework.view.JxlsView"* />  <bean id=*"fileView"* class=*"com.wsc.framework.view.FileView"* /> | **JSON, 엑셀, 첨부파일 등의 View 정의** |
| <bean id=*"messageSource"* class=*"org.springframework.context.support.ReloadableResourceBundleMessageSource"*>  <property name=*"basenames"*>  <list><value>classpath:/messages/message</value></list>  </property>  <property name=*"defaultEncoding"*><value>UTF-8</value>  </property>  </bean> | **메시지 리소스 설정** |
| <bean id=*"localeResolver"* class=*"org.springframework.web.servlet.i18n.SessionLocaleResolver"*>  <property name=*"defaultLocale"* value=*"ko"* />  </bean> | **세션 로케일 설정** |
| <mvc:interceptors>  <bean class=*"com.wsc.common.security.SecurityInterceptor"* />  <bean class=*"org.springframework.web.servlet.i18n.LocaleChangeInterceptor"*>  <property name=*"paramName"* value=*"lang"* />  </bean>  </mvc:interceptors> | **Security 처리를 위한 Interceptor 정의** |
| <import resource=*"wsc-mybatis.xml"* /> | **Mybatis 관련 설정파일 Import** |

1. **wsc-mybatis.xml**

* app.properties에 정의된 Database 연결정보를 사용하여 실제 Mybatis 에서 사용한 Database Connection의 DataSource를 설정하고, 트랜젝션에 대한 정보를 정의하는 설정파일이다.
* 현 프레임워크에서 DAO 는 개별로 생성하여 사용하지 않으며, 공통의 DAO 내에서 모든 처리를 담당한다.
* 만약 여러 개의 Database 가 연동되어야 할 경우 아래 정의한 Set 이 한 Set 더 추가되어야 하며, 개발시에 해당 DAO를 사용할 수 있도록 프레임워크의 변경이 필요하다.

|  |  |
| --- | --- |
| 항목 | 설명 |
| <bean id=*"dataSource"* class=*"org.apache.commons.dbcp.BasicDataSource"*>  <property name=*"driverClassName"* value=*"#{app['jdbc.driver']}"*/>  <property name=*"url"* value=*"#{app['jdbc.url']}"*/>  <property name=*"username"* value=*"#{app['jdbc.user']}"*/>  <property name=*"password"* value=*"#{app['jdbc.pwd']}"*/>  <property name=*"maxActive"* value=*"100"*/>  <property name=*"maxWait"* value=*"1000"*/>  <property name=*"poolPreparedStatements"* value=*"true"*/>  <property name=*"defaultAutoCommit"* value=*"true"*/>  <property name=*"validationQuery"* value=*" SELECT 1 FROM DUAL"* />  </bean> | **DataSource 설정** |
| <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>  <property name=*"dataSource"* ref=*"dataSource"*/>  <property name=*"configLocation"* value=*"classpath:mybatis.xml"* />  <property name=*"mapperLocations"* value=*"classpath:mappers/\*\*/\*.xml"*/>  </bean> | **SQL Session 및 Mybatis 연관 SQL을 담은 XML의 경로를 설정** |
| <bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>  <property name=*"dataSource"* ref=*"dataSource"* />  </bean>  <tx:annotation-driven transaction-manager=*"transactionManager"* /> | **트랜젝션 관련 설정** |
| <bean id=*"commonDao"* class=*"com.wsc.common.dao.CommonDao"*>  <property name=*"sqlSessionFactory"* ref=*"sqlSessionFactory"* />  </bean> | **공통으로 사용할 DAO 설정** |

1. **wsc-jasper.xml**

* iReport 생성시 다음과 리포트 파일과 연결할 설정정보를 정의한다.

Controller 에서 사용할 View Name을 bean 의 id 로 정의 후 Controller에서 해당 View Name 을 사용하면 연동한 iReport의 Report가 웹 상에 표시된다.

|  |
| --- |
| 코드 리포트의 샘플 예제 |
| <bean id=*"codeReport"* class=*"org.springframework.web.servlet.view.jasperreports.JasperReportsMultiFormatView"*>  <property name=*"url"* value=*"/WEB-INF/reports/report5.jasper"*/>  <property name=*"exporterParameters"*>  <map>  <entry key=*"net.sf.jasperreports.engine.JRExporterParameter.CHARACTER\_ENCODING"*><value>UTF-8</value></entry>  <entry key=*"net.sf.jasperreports.engine.export.JRHtmlExporterParameter.IS\_USING\_IMAGES\_TO\_ALIGN"*><value>false</value></entry>  </map>  </property>  </bean> |

## 프레임워크 명명 규칙

1. **패키지 명명 규칙**

* 가장 최상위 패키지는 com.wsc를 기본으로 한다.
* 다음의 패키지는 프레임워크 관련 기본 패키지 이다.

com.wsc.framework: 프레임워크 영역

com.wsc.common: 공통 시스템관리 영역

* 프로젝트의 업무 관련 기본 패키지를 생성 후 업무영역 별로 별도의 패키지를 생성하여 관리하도록 한다. 예를 들어, 업무패키지를 business 라 명명하고, 그 하위에 업체관리, 발급관리를 둘 경우 다음과 같은 패키지가 생성된다.

업무의 기본 패키지: com.wsc.business

업무 내 업체관리 패키지: com.wsc.business.company

업무 내 발급관리 패키지: com.wsc.business.issue

1. **자바리소스 명명 규칙**

* 자바의 리소스는 src/main/resources 폴더 안에 위치하도록 한다.
* 각종 properties 파일 및 메시지 리소스, Mybatis XML이 위치하는 곳이다.

1. **자바 소스 명명 규칙**

* 자바소스는 기본 업무 패키지 안에 위치하며 해당 명명규칙은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| 기능구분 | 명명규칙 | 게시판(Board) 예제 |
| Controller | com.wsc.{업무패키지}.{업무명칭}Controller | om.wsc.common.BoardController |
| Service | com.wsc.{업무패키지}.{업무명칭}Service | com.wsc.common.BoardService |
| Model | com.wsc.{업무패키지}.{업무명칭} | com.wsc.common.Board |

* 위의 경우에서 Model은 필요한 경우에만 생성하며, 일반적으로 RecordMap 을 사용한다.

1. **Mybatis 명명 규칙**

* Mybatis 의 SQL Query를 담은xml은 src/main/resources/mappers안에 위치하도록 한다.
* 경로는 자바클래스의 패키지 경로와 동일하게 설정하며 해당 명칭은 클래스의 기본명칭과 일치시킨다. 예를 들어, 게시판의 경우 기본 패키지 경로는 com.wsc.comon.board 이며, 해당 서비스클래스는 BoardService 이므로 연관된 Mybatis xml 은 {자바리소스폴더}/mappers/com/wsc/common/bard/Board.xml 이 된다.
* Mybatis SQL 파일내의 명명규칙은 다음과 같다.

|  |
| --- |
| 1. namespace=”{업무패키지경로}.{업무클래스 기본명칭}” 2. 검색: search, 검색카운트: searchCount 3. 조회: select 4. 등록: insert 5. 수정: update 6. 삭제: delete 7. parameterType=”params” (com.wsc.framework.model.ParamsMap 의 약칭) 8. 검색/조회 resultType=”record”(com.wsc.framework.model.RecordMap 의 약칭) 9. 위의 기본 SQL 이외에 별도의 SQL 구현이 필요한 경우 기능에 맞는 명칭 뒤에 별도의 확장된 업무 명칭을 추가하여 사용하도록 한다. 예를 들어, 게시판의 게시대상(Target) 검색이 필요한 경우 search 뒤에 Target을 추가하여 “searchTarget” 의 ID로 select 구문을 생성하면 된다.   <select id=*"searchTarget"* resultType=*"record"* parameterType=*"params"*></select> |
| <?xml version=*"1.0"* encoding=*"UTF-8"* standalone=*"no"*?>  <!DOCTYPE mapper  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  <mapper namespace=*"com.wsc.common.board.Board"*>  <select id=*"search"* resultType=*"record"* parameterType=*"params"*></select>  <select id=*"searchCount"* resultType=*"int"* parameterType=*"params"*></select>  <select id=*"select"* resultType=*"record"* parameterType=*"params"*></select>  <insert id=*"insert"* parameterType=*"params"*></insert>  <update id=*"update"* parameterType=*"params"*></update>  <delete id=*"delete"* parameterType=*"params"*></delete>  </mapper> |

1. **URL 및 확장자 규칙**

* 일반적으로 화면을 이동하는 URL 의 확장자는 \*.do를 기본으로 하며, Ajax 통신을 통하여 결과를 받는 URL의 확장자는 \*. json을 기본으로 한다. 해당 확장자에 따라 에러 처리방식이 달라지므로 이 기본 규칙을 반드시 준수하도록 한다.
* 기본 CRUD를 처리하는 URL 의 명명규칙은 다음과 같으며, 이는 각 업무의 Controller 클래스에 RequestMapping 이라는 Annotation에 정의되어 있다.

|  |  |  |
| --- | --- | --- |
| 기능구분 | 명명규칙 | 게시판(Board) 예제 URL |
| 기본 URL | /{업무패키지명 소문자}/{업무명 소문자} | /common/board |
| 관리화면 오픈 | /{업무명 소문자}.do | /common/board/board.do |
| 폼 편집화면 오픈 | /form.do | /common/board /form.do |
| 폼 조회화면 오픈 | /view.do | /common/board /view.do |
| JSON 검색처리 | /search.json | /common/board /search.json |
| JSON 조회처리 | /select.json | /common/board /select.json |
| JSON 저장처리 | /save.json | /common/board /save.json |
| JSON 삭제처리 | /delete.json | /common/board /delete.json |
| 엑셀 다운로드 | /download.do | /common/board /download.do |
| **게시판 Controller Class 예제** | | |
| @Controller  @RequestMapping("/common/board")  **public** **class** BoardController **extends** BaseController {  @RequestMapping(value = "/board.do")  **public** String open(HttpServletRequest request, Model model) {  }  @RequestMapping(value = "/form.do")  **public** String openForm(HttpServletRequest request, Model model) {  **return** getViewName("boardForm");  }  @RequestMapping(value = "/view.do")  **public** String openView(HttpServletRequest request, Model model) {  **return** getViewName("boardView");  }  @RequestMapping(value = "/search.json")  **public** String search(HttpServletRequest request, Model model) {  }  @RequestMapping(value = "/download.do")  **public** String download(HttpServletRequest request, Model model) {  }  @RequestMapping(value = "/select.json")  **public** String select(HttpServletRequest request, Model model) {  }  @RequestMapping(value = "/save.json")  **public** String save(HttpServletRequest request, Model model) {  }  @RequestMapping(value = "/delete.json")  **public** String delete(HttpServletRequest request, Model model) {  }  } | | |

1. **JSP 및 자바스크립트 명명 규칙**

* JSP는 src/main/webapp/WEB-INF/views가 기본 시작경로로 설정되어 있으며 그 하위에 {업무패키지명}/{업무명}/ 안에 정의된다.
* JSP 의 경로 및 명칭은 Controller 클래스에서 정의한 View Name 과 밀접하게 연관되어 있다. 5) 번의 명명규칙 내에서 보면 관리화면 오픈을 하는 URL의 경우 “{업무명 소문자}.do”를 기본으로 하고 있는데 이 명칭이 곧 JSP 의 명칭과 맵핑된다. 단, 명칭의 변경이 필요할 경우 getViewName 이라는 메서드를 호출하여 해당 메서드에 인자로 JSP의 확장자를 제외한 명칭을 넘겨주면, 해당 명칭으로 맵핑될 수 있다. 예를 들어, 게시판의 관리화면 오픈은 기본이기 때문에 board.jsp 라는 명칭으로 정의되지만, 폼 편집화면 오픈인 경우 boardForm.jsp로 JSP를 명명하고 BoardController 에선 getViewName(“boardForm”) 이라는 메서드를 사용하여 맵핑하는 것을 볼 수 있다.
* 자바스크립트는 src/main/webapp/resources/js 안에 업무패키지 및 업무명칭 별로 정의된다.
* 기본적으로 CRUD 하나의 Set을 개발하기 위해서는 다음의 소스들이 생성되어야 한다.

1. Spring MVC Controller Class

: @Controller 라는 Annotation 이 명기되어야 한다.

: @RequestMapping 을 통해 URL이 명시되어야 한다.

: JSP 와 연동되는 View Name 이 명시되거나 클래스의 기본명칭과 JSP 명칭이 일치되어야 한다.

: BaseController를 상속받아야 한다.

1. Spring Service Class

: @Service 라는 Annotation이 명기되어야 한다.

: 트랜젝션 처리가 필요한 경우 @Transaction 을 처리가 필요한 메서드에 명기해야 한다.

: BaseService를 상속받아야 한다.

1. Mybatis XML include SQL

: 연동할 Service 의 패키지를 포함한 기본명칭과 namespace를 일치시켜야 한다.

: “search”, “searchCount”, “select”, “insert”, “update”, “delete” 의 기본 SQL ID에 해당하는 구문을 가급적 모두 작성하도록 한다.  
: 해당 SQL ID를 통해 Service Class 에서 사용된다.

1. JSP

: Controller Class 에서 정의한 View Name 또는 기본명칭과 일치하는 JSP 를 생성한다.

: JSP의 기본 틀은 소스를 참고하도록 한다.

: 하나의 업무 스크립트를 반드시 INCLUDE 하도록 구현한다.

: EasyUI 의 API를 참고하여 구현하도록 한다.

1. JavaScript

: 기본적으로 jQuery 기반의 스크립트가 작성되어야 하며, EasyUI를 활용한 기능구현이 필요하므로 EasyUI API를 항시 참고하도록 한다.

위의 5가지의 파일들이 명명규칙에 맞게 구현이 되면 하나의 관리 기능이 완성된다.

## 자바소스 파일목록

|  |  |
| --- | --- |
| Class | Description |
| com.wsc.framework.base.BaseComponent  com.wsc.framework.base.BaseConstants  com.wsc.framework.base.BaseController  com.wsc.framework.base.BaseDao  com.wsc.framework.base.BaseException  com.wsc.framework.base.BaseInterceptor  com.wsc.framework.base.BaseMap  com.wsc.framework.base.BaseModel  com.wsc.framework.base.BaseResultHandler  com.wsc.framework.base.BaseService  com.wsc.framework.base.BaseView | 프레임워크 공통 기본 클래스 |
| com.wsc.framework.model.PagingMap  com.wsc.framework.model.ParamsMap  com.wsc.framework.model.RecordMap  com.wsc.framework.model.ResultMap | 프레임워크 공통 Model Map |
| com.wsc.framework.exception.AuthorityException  com.wsc.framework.exception.ServiceException  com.wsc.framework.exception.SessionException  com.wsc.framework.exception.SystemException | 프레임워크 공통 Exception |
| com.wsc.framework.view.DownloadView  com.wsc.framework.view.FileView  com.wsc.framework.view.JxlsView | 프레임워크 공통 사용 View |
| com.wsc.framework.excel.ExcelLoader | 프레임워크 엑셀로더 |
| com.wsc.framework.utils.BaseUtils  com.wsc.framework.utils.FileUtils  com.wsc.framework.utils.MybatisUtils | 프레임워크 공통 유틸 모음 |
| com.wsc.common.dao.CommonDao | 프레임워크 공통 DAO |
| com.wsc.common.security.CommonController  com.wsc.common.security.SecurityInterceptor  com.wsc.common.security.SessionComponent | 프레임워크 인증 및 권한 처리 모듈 |
| com.wsc.common.Consts | 시스템관리 업무의 공통 상수 |
| com.wsc.common.model.Code  com.wsc.common.model.FileInfo  com.wsc.common.model.Group  com.wsc.common.model.Menu  com.wsc.common.model.Program  com.wsc.common.model.User | 프레임워크 공용 모델 |
| com.wsc.common.board.AddressController  com.wsc.common.board.AddressService  com.wsc.common.board.BoardController  com.wsc.common.board.BoardGroup  com.wsc.common.board.BoardService  com.wsc.common.board.EmailController  com.wsc.common.board.ReplyController | 일반게시판, 답변게시판, 전자메일 등의 게시판 유형 구현 모듈 |
| com.wsc.common.code.CacheComponent  com.wsc.common.code.CodeController  com.wsc.common.code.CodeService | 공통코드, 공통코드 엑셀로더, 공통코드 캐쉬 기능 구현 모듈 |
| com.wsc.common.file.FileController  com.wsc.common.file.FileDirectory  com.wsc.common.file.FileService | 첨부파일 기능 구현 모듈 |
| com.wsc.common.user.GroupController  com.wsc.common.user.GroupService  com.wsc.common.user.MenuController  com.wsc.common.user.MenuService  com.wsc.common.user.ProgramController  com.wsc.common.user.ProgramService  com.wsc.common.user.UserController  com.wsc.common.user.UserLogController  com.wsc.common.user.UserLogService  com.wsc.common.user.UserSecureService  com.wsc.common.user.UserService | 사용자, 사용자그룹, 화면, 메뉴, 권한관리 등의 기능 구현 모듈 |
| com.wsc.common.schedule.ScheduleController  com.wsc.common.schedule.ScheduleGroup  com.wsc.common.schedule.ScheduleService | 일정관리 기능구현 모듈 |
| com.wsc.common.report.ReportController | 리포트 기능구현 모듈 |

## 스크립트 공통파일

|  |  |  |
| --- | --- | --- |
| 경로 | 파일 및 폴더 | 설명 |
| src/main/webapp/resources/jquery/ | easyui-1.4 | 화면UI 프레임워크 |
| uploader-3.1 | 파일 업로드 Plugin |
| fullcalendar-2.1.1  jquery.schedule.js | 일정관리 Plugin |
| sceditor-1.4.5  ckeditor-4.4.7  daumeditor-7.5.9 | 웹에디터 Plugin |
| jquery-ui-1.11.4 | jQuery UI CSS |
| src/main/webapp/resources/js/include/ | utilities.js | 기본 유틸 함수 모음 |
| extension.js | 문자열 확장 함수 모음 |
| common.js | 공통 기능 지원 함수 모음  (모든 화면에서 초기호출 함수가 있음) |
| widget.js | 공통적으로 사용되는 상단메뉴,좌측메뉴,사이트맵 등의 위젯 함수 모음 |
| business.js | 업무로직 구현에 필요한 기능 함수 모음  그리드,폼,다이얼로그 등의 기능 함수 |
| system.js | 시스템관리에 공통으로 사용되는기능 모음  (사용자목록,그룹목록,일정공유 등) |
| src/main/webapp/resources/js/module/ | jboard.js | 게시판 공통 기능 |
| jeditor.js | 웹에디터 공통 기능 |
| jupload.js | 파일업로드 공통 기능 |

# 기능별 설명

## Code Cache 설정

* Code를 Caching 하기 위해 Ehcache 라는 라이브러리가 사용되었으며, 해당 설정은 ehcache.xml 에 정의되어 있다.
* 자세한 사용법은 ehcache 라이브러리의 URL을 참고하도록 한다.
* Code Cache를 위해서 CodeService 의 searchCodes 메서드에 Annotation 으로 @Cacheable 이라고 선언되어 있으며, save, saveExcel 메서드에는 @CacheEvict 이 선언되어 있는 것을 볼 수 있다.
* 기본적으로 검색 시엔 최초 모든 Code 목록을 가져오며 이를 Cache 에 저장해 두었다가, 호출시 별도의 Database 검색을 거치지 않고 Caching 된 코드 목록이 반환된다.
* 만약 공통코드 쪽의 저장행위가 일어나면 Caching 되었던 코드가 모두 Reset 되며, 이후 searchCodes 가 호출될 때 다시 Database 에서 해당 코드가 검색되고 Cache 저장소에 담긴다.

## 시스템 ID 설정

* 현재의 프레임워크에선 기본적으로 SYS\_ID 라는 시스템ID 가 항상 사용된다.
* 시스템 ID는 app.properties 의 default.system.id 의 항목에 Default 값이 선언되어 있다.
* 현재는 Default로 선언된 시스템 ID를 사용하나, 사용자 로그인시마다 별도의 시스템 ID를 선택할 수 있도록 구현되어 있기도 하다. 다만, 이는 감춰져 있으며 이를 사용하고자 할 경우엔 사용자 로그인시에 해당 시스템ID를 선택할 수 있도록 콤보 형태로 변형하여 구현할 필요가 있다.
* 다만 주의해야 할 것은 Database 에서 검색하는 모든 항목은 기본적으로 시스템 ID가 설정되어 있으므로, 반드시 인증 후에 시스템의 이용이 가능한 업무시스템에서 선택사항을 적용해야 한다.

## 인증 및 권한 설정

* 인증 및 권한 처리를 담당하는 클래스는 SecurityInterceptor 이다. 내부적으로 사용되는 변수는 app.properties에 정의되어 있는 인증 및 권한 처리와 관련된 URL 및 인증에 필요한 Parameter의 명칭을 담고 있다.
* SecurityInterceptor를 사용하기 위한 설정은 wsc-context.xml 에 정의되어 있으며, 해당 URL을 처리하는 Controller는 CommonController 에 정의되어 있다.
* Spring 의 Servlet 을 거치는 모든 URL은 항상 SecurityInterceptor의 preHandle 메서드를 최우선으로 실행하게 되며, 실행 결과에 따라 해당 Controller의 View 결과를 실행하게 될지 결정된다.
* 다음은 preHandle의 실행절차를 설명한 것이다.

1. 기본적인 사용자의 접근정보를 Request에 담는다.

(browser, context. ip, uri)

1. URL 이 로그인(LOGIN) 페이지인 경우 해당 페이지로 Forward.
2. URL 이 인증처리(SUBMIT) 페이지인 경우 인증처리(processLogin) 실행

: 인증요청 정보에 맞는 사용자가 있는지 확인 및 없을 경우 에러처리

: 관리자가 사용자로 자동 로그인을 실행한 경우인지 확인 후 해당 로그인 처리 (secureKey 확인)

: 인증 실패시 해당 사용자의 실패 카운트 증가 및 에러처리.

: 인증 성공시 해당 사용자의 최종로그인 일시 업데이트 및 실패 카운트 리셋 후 세션정보 저장.

1. URL 이 로그아웃(LOGOUT) 페이지인 경우 세션 제거 후 로그인 페이지로 Redirect.
2. URL이 위의 항목들을 포함하지 않는 경우 다음의 프로세스로 진행.

: 관리하는 화면정보가 아니면 SKIP.

: 인증 사용자의 경우 화면권한정보 확인.

: 인증 사용자의 경우 사용자세션정보를 Request 에 담는다.

(sysId, userId, userName, admin, pageSize)

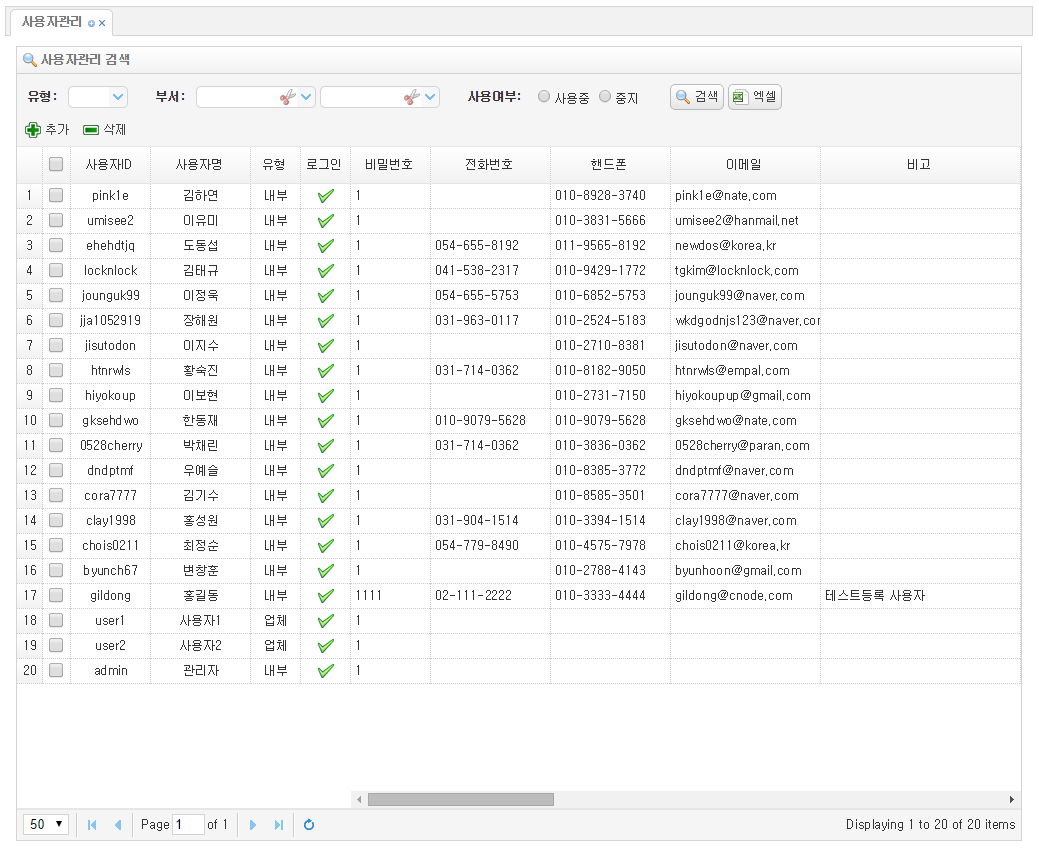
: 화면정보가 있는 경우 Request에 담는다.

(progId, progName)

: 사용자 접근 DB 로깅 사용(app.propertis 의 loggable.userLog참고)시 해당 접근정보를 DB(SYS\_ULOG)에 저장한다.

: 화면접근이 제한된 사용자인 경우 에러처리

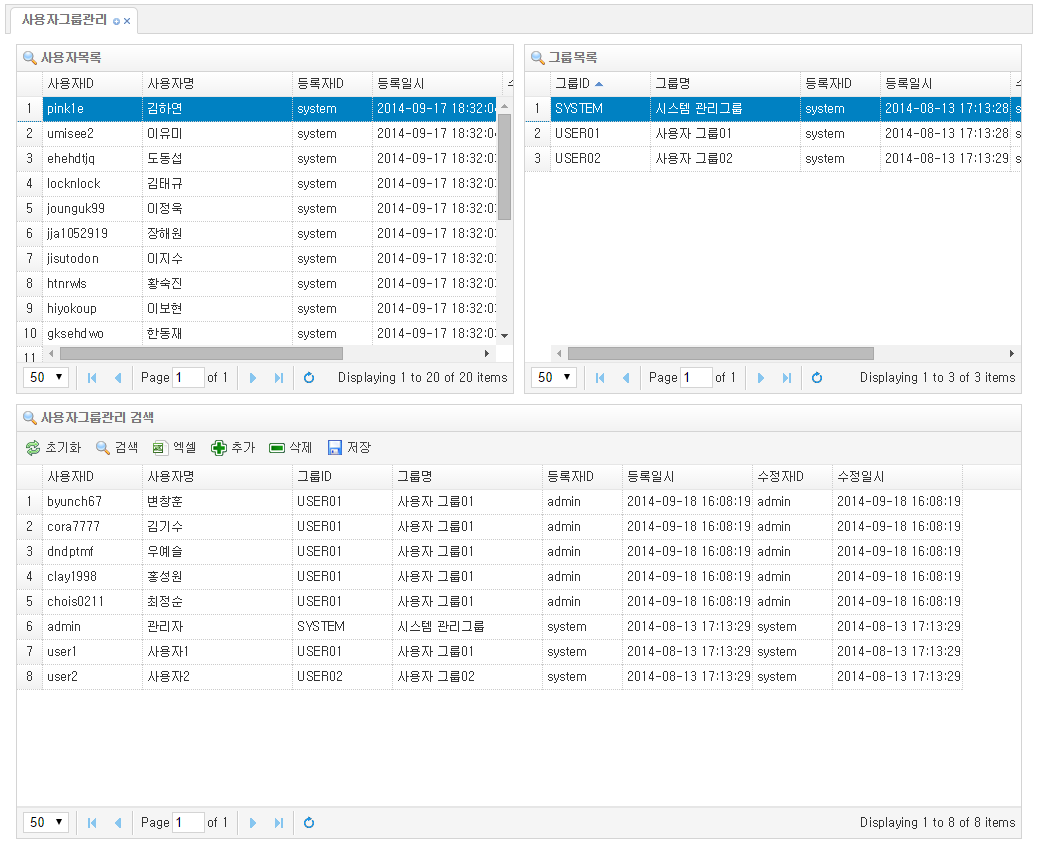
## 관리자의 자동 로그인 기능

* 관리자의 자동로그인 기능이란 관리자가 사용자를 대신하여 시스템 상의 업무를 대행해주거나, 문제의 원인을 분석하고자 할 때 사용하는 기능으로서, 사용자의 세션으로 자동 로그인되어 시스템을 해당 사용자와 동일한 화면으로 접근할 수 있게 해주는 것이다.
* 이는 관리자로 로그인 후에 시스템관리 – 사용자관리 화면으로 접속하여 사용할 수 있다.
* 이 기능을 활성화하기 위해서는 관리자와 사용자의 세션이 생성될 수 있도록 서버의 설정이 반드시 필요하다. 현 시스템에 대해 두 개의 도메인이 존재해야 하며, 사용자가 접근하는 도메인과 관리자가 접근하는 도메인이 달라야 한다. 이에 대한 서버의 설정이 이루어지면 **app.properties 의 domain.user** 에 해당 도메인을 명시해주어야 한다.

아이콘 클릭

## 사용자 그룹

* 사용자 그룹은 현재 기준으로 SYSTEM(시스템관리그룹), USER01(사용자그룹01), USER02(사용자그룹02) 의 3가지로 되어 있으나, 이 중 SYSTEM 은 반드시 필요한 그룹이다.
* SYSTEM 그룹은 반드시 정의되어야 하며, 이는 어플리케이션 상에 ADMIN\_GROUP 으로 정의되어 있다.
* ADMIN\_GROUP 에 추가하고자 하는 그룹이 있다면 **BaseConstants.ADMIN\_GROUPS** 에 추가할 그룹 ID를 add하도록 한다.
* SYSTEM 그룹은 반드시 정의되어야 하며, 이는 어플리케이션 상에 ADMIN\_GROUP 으로 정의되어 있다.
* 사용자그룹은 시스템관리 – 사용자그룹관리를 사용하여 사용자와 그룹을 맵핑한다.



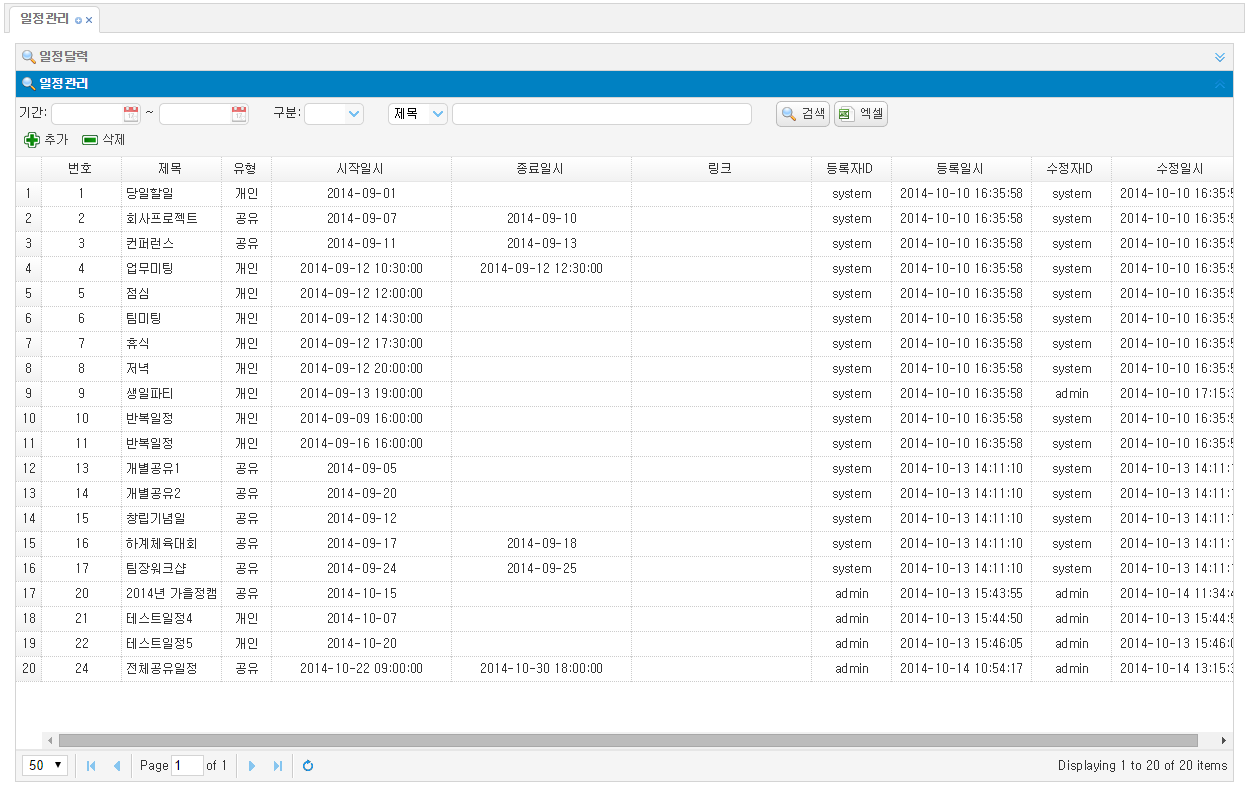
## 화면 및 메뉴 권한 설정

* 화면에 대한 접근 권한의 제한 처리는 관리하고 있는 화면 목록을 기준으로 한다. 따라서 관리되지 않는 화면까지 모든 제한 처리가 필요한 경우엔 SecurityInterceptor 의 변경이 요구된다.
* 메뉴의 권한은 동일한 URL을 기준으로 해당 화면의 권한이 있는지를 체크하여 활성화,비활성화 형태로 표시된다.
* 기본 CRUD 를 구현 후 해당 화면의 접근권한을 설정해야할 경우에 사용되며, 시스템관리의 [화면관리],[메뉴관리],[그룹화면관리],[사용자화면관리] 의 기능을 이용하여 권한을 설정한다.
* 이에 대한 처리는 샘플게시판 개발 가이드의 권한 설정 부분에 설명되어 있으므로 별도의 기능설명은 생략한다. 다만, 사용자화면관리는 그룹화면에 정의되어 있는 부분 이외에 예외사항을 처리하기 위한 용도이므로 특수한 경우가 아니면 별도의 관리가 필요없다.

## Pagination 기능

* 페이징은 그림과 같이 Grid에 Pagination 속성이 정의되어 있으면 표시되며, 현 시스템에선 Server Side Pagination 을 기본으로 한다. 따라서, 해당 속성이 화면상에 정의될 경우 페이징에 관한 정보가 다음의 Parameter 들이 전달되며, Java Source 상에선 해당 Parameter 가 있을 경우 페이징된 결과 목록을 반환한다.

|  |  |  |
| --- | --- | --- |
| Type | Parameter Name | Description |
| 페이징 요청 Parameter | page | 현재 페이지 번호 |
| rows | 현재 화면당 출력수 |
| 검색 결과 페이징 데이터 | total | 검색한 총 건수 |
| pages | 검색한 총 페이지수 |
| count | 현재 결과 목록 건수 |
| rows | 검색한 결과 목록 데이터 |



Pagination 영역

* 페이징은 자동으로 프레임워크에서 처리되어 반환되므로 자바 소스 단위에선 별도의 구현은 필요 없으며, 다만 Mybatis 의 SQL을 구현할 때 반드시 검색 SQL 이외에 Count SQL이 필요하고, 페이징은 위한 SubQuery 가 구현되어야 한다.

|  |
| --- |
| 페이징 구현 예제 |
| <sql id=*"pagingColumns"*>  페이징에 필요한 칼럼  <if test=*"start == null or end == null"*>  ,0 AS "rnum"  </if>  </sql>  <sql id=*"pagingOpen"*>  <if test=*"start != null and end != null"*>  페이징 시작 구문  SELECT \*  FROM (  SELECT X.\*  ,ROWNUM as "rnum"  FROM (  </if>  </sql>  <sql id=*"pagingClose"*>  <if test=*"start != null and end != null"*>  ) X  페이징 종료 구문  WHERE ROWNUM <![CDATA[< ]]> #{end}  )  WHERE "rnum" <![CDATA[>= ]]> #{start}  </if>  </sql>  <sql id=*"sortClause"*></sql>  정렬 구문  <sql id=*"selectColumns"*></sql>  검색 칼럼  <sql id=*"whereClause"*></sql>  검색 조건 절  <select id=*"search"* resultType=*"record"* parameterType=*"params"*>  <include refid=*"pagingOpen"* /><!-- PAGING -->  SELECT <include refid=*"selectColumns"* />  <include refid=*"pagingColumns"* /><!-- PAGING -->  FROM SYS\_BORD A  WHERE <include refid=*"whereClause"* />  ORDER  BY <include refid=*"sortClause"* /><!-- SORT -->  <include refid=*"pagingClose"* /><!-- PAGING -->  </select>  <select id=*"searchCount"* resultType=*"int"* parameterType=*"params"*>  SELECT COUNT(1)  FROM SYS\_BORD A  WHERE <include refid=*"whereClause"* />  </select> |

## Web Editor 기능

* 현 시스템의 웹 에디터는 다음의 3가지 플러그인을 제공하고 있으며, 해당 플러그인을 jeditor.js 에서 선택하여 사용 가능하며, 게시판 소스를 참고하여 사용하도록 한다.

1. SC: Sceditor jQuery Plugin : /src/main/webapp/resources/jquery/sceditor-1.4.5
2. DM: 다음에디터 Plugin: /src/main/webapp/resources/jquery/daumeditor-7.5.9
3. CK: CKEditor jQuery Plugin: /src/main/webapp/resources/jquery/ckeditor-4.4.7

* jeditor.js 스크립트 내의 jeditor\_consts 변수값을 위의 3가지 중 하나의 약어로 설정하면 해당 플러그인이 에디터로 적용된다. 단, 현재의 웹에디터는 IE8 이하로는 실행이 불가하다. 자세한 API 는 해당 Plugin 사이트에서 확인하도록 한다.
* 다음은 개발시 웹에디터 기능이 필요한 경우의 절차이다.

1. 사용할 화면의 JSP 의 textarea 태그에 다음과 같이 class 를 설정한다.

<textarea name=*"bordText"* id=*"r\_bordText"* class=*"sceditor wui-editor"*></textarea>

1. 사용할 화면의 업무 스크립트에 다음의 코드를 추가한다.

using("../../js/module/jeditor.js");

1. 업무 스크립트에서 초기화 함수에 다음과 같이 Editor 객체를 생성한다.

**var** editorObj **= new** jeditor({

editorLayer: "editor-area", => 웹에디터 레이어 ID

editorBox: "r\_bordText", => 웹에디터 텍스트박스 ID

editorForm: "regist-form" => 웹에디터 폼 ID

});

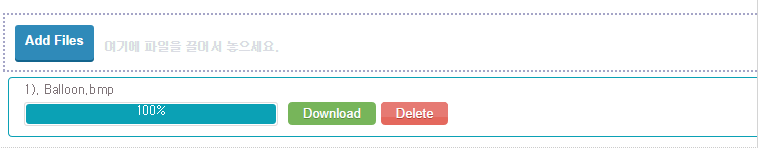
1. 업무 스크립트에서 저장 시 해당 값을 가져오기 위해서 다음과 같이 사용한다.

**var** textData **=** editorObj.get();

1. 업무 스크립트에서 수정 시 해당 값을 textarea 에 바인딩하기 위해서 다음과 같이 사용한다.

editorObj.set( textData );

## 첨부파일 Control 기능

* 현 시스템의 첨부파일 처리는 Upload File jQuery Plugin 을 사용하고 있으며 /src/main/webapp/resources/jquery/uploader-3.1 에 INCLUDE하고 있다. 이는 다중파일 업로드 시에 사용되는Plugin 으로, 단순히 단일 파일 하나를 업로드 하기 위해선 별도의 Plugin 없이 사용하도록 한다. 현재 다중파일 업로드는 게시판 영역에 구현되어 있으며, 단일 파일 업로드는 코드엑셀로더를 참고하도록 한다.
* **다중파일 업로드 기능 추가 방법** 

1. FileDirectory 클래스에 추가할 문서그룹에 해당되는 경로정보를 하나 추가한다.

***BOARD\_EMAIL*** ("/board07", "B07", "전자메일")

1. Service 클래스에 저장 시 첨부파일 정보가 저장될 수 있도록 다음과 같은 코드를 추가한다. params 에는 반드시 “atchGrup” 와 “atchNo” 가 존재해야 한다.

fileService.saveFile(params);

1. 사용할 화면의 JSP 의 업로드 컨트롤 표시영역에 DIV 태그를 설정하며 id는 원하는 대로 설정한다.

<div id=*"fileupload"*></div>

<input type=*"hidden"* name=*"atchGrup"* id=*"r\_atchGrup"* value=*"*${model.atchGrup}*"*/>

<input type=*"hidden"* name=*"atchNo"*  id=*"r\_atchNo"*  value=*"*${model.atchNo}*"*/>

1. 사용할 화면의 업무 스크립트에 다음의 코드를 추가한다.

using('../jquery.upload.js');

1. 업무 스크립트에서 초기화 함수에 다음과 같이 upload 객체를 생성한다. 객체 생성시 업로드인지 상세인지를 구분하도록 한다.

**var** uploadObj = jupload.init({

mode: "regist", //업로드: "regist", 검색: "select"

layoutKey: "fileupload", //파일업로드영역 ID (필수)

fileName: "file", //파일박스 NAME

gridKey: "uploaded-files", //파일그리드 ID (필수)

loaderKey: "uploadfile", //파일업로드박스 ID

loaderText: "Add Files" //파일업로드버튼 명칭

}, {

atchGrup:$(*"#r\_atchGrup"*).val(),//첨부파일 문서유형

atchNo: $(*"#r\_atchNo"*).val(), //첨부파일 문서번호

});

1. 업무 스크립트에서 저장 시 해당 값을 가져오기 위해서 다음과 같이 사용한다.

**var** files **=** uploadObj.upload.getFiles();

* **단일파일 업로드 기능 추가 방법**

1. FileDirectory 클래스에 추가할 문서그룹에 해당되는 경로정보를 하나 추가한다.

***BOARD\_EMAIL*** ("/board07", "B07", "전자메일")

1. Controller 클래스에서 업로드 실행 메서드에 다음과 같이 코드를 입력한다.

MultipartFile file = request.getFile("atchFile");

**if**(file != **null** && file.isEmpty() == **false**) {

//필요한 파일정보 처리

}

1. Service 클래스에 저장 시 첨부파일 정보가 저장될 수 있도록 다음과 같은 코드를 추가한다. params 에는 반드시 “atchGrup” 와 “atchNo” 가 존재해야 한다.

fileService.saveFile(params);

1. 사용할 화면의 JSP 의 업로드 컨트롤 표시영역에 다음과 같이 필요한 Input Box를 추가한다.

<div id=*"fileupload"*></div>

<input type=*"file"* name=*"atchFile"* id=*"r\_atchFile"* size=*"40"*/>

<input type=*"hidden"* name=*"atchGrup"* id=*"r\_atchGrup"* value=*"*${model.atchGrup}*"*/>

<input type=*"hidden"* name=*"atchNo"*  id=*"r\_atchNo"*  value=*"*${model.atchNo}*"*/>

1. 사용할 화면의 업무 스크립트에서 업로드 Action 에 다음과 같이 ajaxForm 을 사용하여 업로드 및 저장을 수행한다.s

$('#regist-form').ajaxForm({

url: **this**.url.upload,

beforeSubmit: **function** () {

**if** ($("#r\_atchFile").val() == '') {

$.messager.alert('Warning','업로드파일을 선택하세요.','warning');

**return** **false**;

}

**return** **true**;

},

//submit이후의 처리

success: **function**(data) {

},

//ajax error

error: **function**() {

}

}).submit();

## Combo Box & Combo Grid 기능

* **콤보 박스 구현 방법**

아래의 예제와 같이 input box 의 class 에 easyui-combobox 를 설정하고 data-options 를 설정하면 간단하게 공통코드를 활용한 콤보박스를 구현할 수 있다. 여기에서 반드시 필요한 부분은 loader 를 jcombo.loader 라고 명시해야 하며, params 항목에 codeGrup 이라는 KEY 가 반드시 설정되어야 한다.

다만, data-options의 params 를 사용하지 않겠다면 input box 의 attribute 로 codeGrup=”USER\_TYPE” 이라고 명시해도 된다. params 를 사용하는 이유는 codeGrup 외에 별도의 조건들이 더 필요할 수 있기 때문이다. code의 extChr01, …, extChr10 등의 선택적인 조건을 더 추가할 수 있다.

<input class=*"easyui-combobox"* name=*"userType"* id=*"s\_userType"*

data-options=*"width:60,*

*mode:'remote',*

*editable:false,*

*loader:jcombo.loader,*

*params:{codeGrup:'USER\_TYPE'}*

*"*

/>

* **단계 콤보 구현 방법**

단계 콤보란 콤보간의 연동이 되어 1단계 선택 시 2단계의 옵션이 바뀌어야 하는 경우 사용되는 기능이다. 이는 일반 콤보를 응용하여 사용되고 있으므로 기본 방식은 일반 콤보와 같다. 다만 상위 콤보에는 child 옵션과 onChange 옵션이 추가되며, 하위 콤보에는 parent 옵션이 추가된다.

<input class=*"easyui-combobox"* name=*"deptPrnt"* id=*"s\_deptPrnt"*

data-options=*"width:120,*

*mode:'remote',*

*loader:jcombo.loader,*

*params:{codeGrup:'DEPT\_CODE',extNum01:1},*

*child: {id:'s\_deptCode',name:'deptCode'},*

*onChange: jcombo.select*

*"*

/>

<input class=*"easyui-combobox"* name=*"deptCode"* id=*"s\_deptCode"*

data-options=*"width:120,*

*mode:'remote',*

*loader:jcombo.loader,*

*params:{codeGrup:'DEPT\_CODE',extNum01:2},*

*parent:{id:'s\_deptPrnt',name:'extChr01'}*

*"*

/>

* **콤보 그리드 구현 방법**

콤보가 그리드 형태로 표현되는 것을 콤보 그리드라고 하며, 이는 EasyUI에서 지원되는 기능이다.

콤보의 class가 easyui-combogrid 로 명시되며 data-options 는 그리드의 옵션들을 설정할 수 있다.

<input class=*"easyui-combogrid"* name=*"userType"* id=*"r\_userType"*

data-options=*"width:100,*

*required:true,*

*editable:false,*

*fitColumns:true,*

*panelWidth:150,*

*textField:'codeName',*

*idField:'codeCd',*

*method:'post',*

*url:'*${context}*/common/code/code.json',*

*queryParams:{'codeGrup':'USER\_TYPE'},*

*onChange: function(newVal, oldVal) {*

*$('#r\_userTypeCd').textbox('setValue', newVal);*

*},*

*columns: [[*

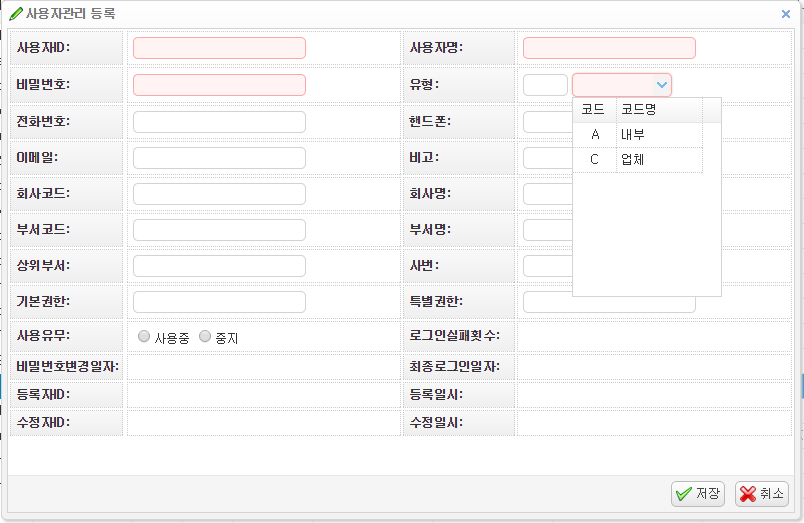
*{field:'codeCd',align:'center',width:50,title:'코드'},*

*{field:'codeName',align:'left',width:100,title:'코드명'}*

*]]*

*"*

/>



## Excel Download 기능

* 엑셀 다운로드는 jxls 라이브러리를 사용하였으며, 버전은 1.0.5 이다. 이 기능은 서버에서 검색한 데이터를 사전 정의된 템플릿과 병합하여 엑셀파일을 생성하여 내려주는 기능이다. 따라서, 엑셀 다운로드를 구현하기 위해선 반드시 해당 화면의 엑셀 템플릿 작업이 필요하다.
* 엑셀 템플릿 경로는 app.properties 에 정의되어 있으며, 현재는 src/main/webapp/WEB-INF/templates 에 위치해 있다. 템플릿 명칭은 기본적으로 해당 Controller 클래스의 기본명칭을 사용한다. 예를 들면, 코드관리의 경우 CodeController 이므로 템플릿 명칭은 Code\_Template.xlsx 이고, 다운로드시의 명칭은 Code.xlsx 가 된다. 이를 변경하고자 한다면 다음과 같이 Controller의 download 메서드에 구현하면 된다.

|  |
| --- |
| 예제 |
| ParamsMap params = getParams(request);  Object result = getService().search(params);  model.addAttribute(JxlsView.***TEMPLATE\_NAME***, "Test\_Template.xlsx");  model.addAttribute(JxlsView.***DOWNLOAD\_NAME***, "Test.xlsx");  model.addAttribute(BaseConstants.***DATA***, result);  **return** "jxlsView"; |

* 엑셀 템플릿의 작성방법은 jxls 라이브러리의 API 를 참고하도록 한다.
* 엑셀 다운로드를 위해서 JxlsView 가 사용된다.
* 다음은 엑셀 다운로드 기능을 구현할 경우의 절차이다.

|  |
| --- |
| 1. 해당 업무의 Controller Class에 다음과 같이 추가한다. 검색 기능은 search 메서드를 기본적으로 사용하므로 만약 검색 메서드가 다르게 구현되어야 할 경우엔 super.download 의 내용을 응용하여 구현하도록 한다. |
| @RequestMapping(value = "/download.do")  **public** String download(HttpServletRequest request, Model model) {  **return** **super**.download(request, model);  } |

|  |
| --- |
| 1. 아래의 예제와 같이 엑셀 템플릿을 엑셀 파일로 생성한다. |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **코드목록** | | | | | | **코드그룹** | **코드** | **코드명칭** | **순서** | **코드설명** | | <jx:forEach var="data" items="${rows}"> | | | | | | ${data.codeGrup} | ${data.codeCd} | ${data.codeName} | ${data.sortSeq} | ${data.codeDesc} | | </jx:forEach> | | | | | |

|  |
| --- |
| 1. 해당 업무 화면의 JSP에 다음과 같이 엑셀 버튼을 추가한다. |
| <a href=*"javascript:void(0)"* class=*"easyui-linkbutton"* iconCls=*"icon-excel"* id=*"excel-button"* >엑셀</a> |

|  |
| --- |
| 1. 해당 업무 화면의 스크립트에 다음과 같이 엑셀 버튼 클릭 이벤트를 추가한다. |
| $(**function**() {  //엑셀버튼 클릭시 이벤트 바인딩  $("#excel-button").bind("click", doExcel);  });  //엑셀 다운로드  **function** doExcel() {  consts.easygrid.download();  } |

## Excel Loader 기능

* 엑셀 파일을 업로드 하여 해당 내용을 라인단위로 읽어서 로드한 후에 화면에서 필요한 부분을 편집 후 저장하는 기능이다. 이는 코드엑셀로더에 구현되어 있으며, jxls 라이브러리의 loader 기능을 사용하고 있다.
* 엑셀 파일 로드시 필요한 항목에 대한 맵핑 설정파일을 xml로 정의하여 이를 읽어 해당 엑셀을 파싱하고 내려주며 이 맵핑설정파일의 위치에 대한 정보는 app.properties에 정의되어 있다. 현재 시스템에선 src/main/resources/xlsloaders/ 에서 확인할 수 있다.
* 현재 구현된 기능은 엑셀파일을 업로드하여 바로 ExcelLoader 를 사용하여 파싱한 후 결과목록을 Json으로 화면에 넘겨주도록 되어 있으며, 따라서 별도로 파일이 서버에 저장되지 않는다. 만약 서버에 저장되도록 하기 위해선 ExcelLoader 의 load 메서드에서 물리적 파일을 저장하는 로직이 구현되어야 한다.
* 엑셀로더 기능을 사용하기 위한 절차는 다음과 같다.

|  |
| --- |
| 1. 해당 업무의 Controller Class에 다음과 같이 3개의 메서드가 추가되어야 한다.  * 엑셀로더 화면 오픈 메서드 * 업로드후에 결과목록을 반환하는 메서드 * 편집된 목록을 받아 저장하는 메서드 |
| //엑셀로더화면오픈  @RequestMapping(value = "/loader/open.do")  **public** String openLoader(HttpServletRequest request, Model model) {  **return** **super**.getViewName("codeLoader");  }  //엑셀 업로드  @RequestMapping(value = "/loader/upload.json")  **public** String uploadLoader(MultipartHttpServletRequest request, Model model) {  //엑셀 업로드  MultipartFile file = request.getFile("excelFile");  **if** (file != **null** && file.isEmpty() == **false**) {  Map result = loader.load(file, "jxls-code.xml");  addObject(model, result.get("list"));  }  **return** "jsonView";  }  //엑셀 저장  @RequestMapping(value = "/loader/save.json")  **public** String saveLoader(HttpServletRequest request, Model model) {  ParamsMap params = getParams(request, **true**);  Object result = service.save(params);  addObject(model, result);  **return** "jsonView";  } |

|  |
| --- |
| 1. 해당 업무의 JSP 와 스크립트를 구현한다. |
| codeLoader.jsp 와 codeLoader.js 파일을 참고할것. |

|  |
| --- |
| 1. 엑셀에 맵핑되는 로더 설정 파일을 구현한다. |
| jxls-code.xml 참고할 것.  <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <workbook>  <worksheet name=*"Sheet1"*>  <section startRow=*"0"* endRow=*"0"*>  <mapping row=*"0"* col=*"0"* >header.codeGrup</mapping>  <mapping row=*"0"* col=*"1"* >header.codeCd</mapping>  <mapping row=*"0"* col=*"2"* >header.codeName</mapping>  <mapping row=*"0"* col=*"3"* >header.sortSeq</mapping>  <mapping row=*"0"* col=*"4"* >header.codeDesc</mapping>  </section>  <loop startRow=*"1"* endRow=*"1"* items=*"list"* var=*"item"* varType=*"com.wsc.framework.model.RecordMap"*>  <section startRow=*"1"* endRow=*"1"*>  <mapping row=*"1"* col=*"0"* >item.codeGrup</mapping>  <mapping row=*"1"* col=*"1"* >item.codeCd</mapping>  <mapping row=*"1"* col=*"2"* >item.codeName</mapping>  <mapping row=*"1"* col=*"3"* >item.sortSeq</mapping>  <mapping row=*"1"* col=*"4"* >item.codeDesc</mapping>  </section>  <loopbreakcondition>  <rowcheck offset=*"2"*>  <cellcheck offset=*"2"*></cellcheck>  </rowcheck>  </loopbreakcondition>  </loop>  </worksheet>  </workbook> |

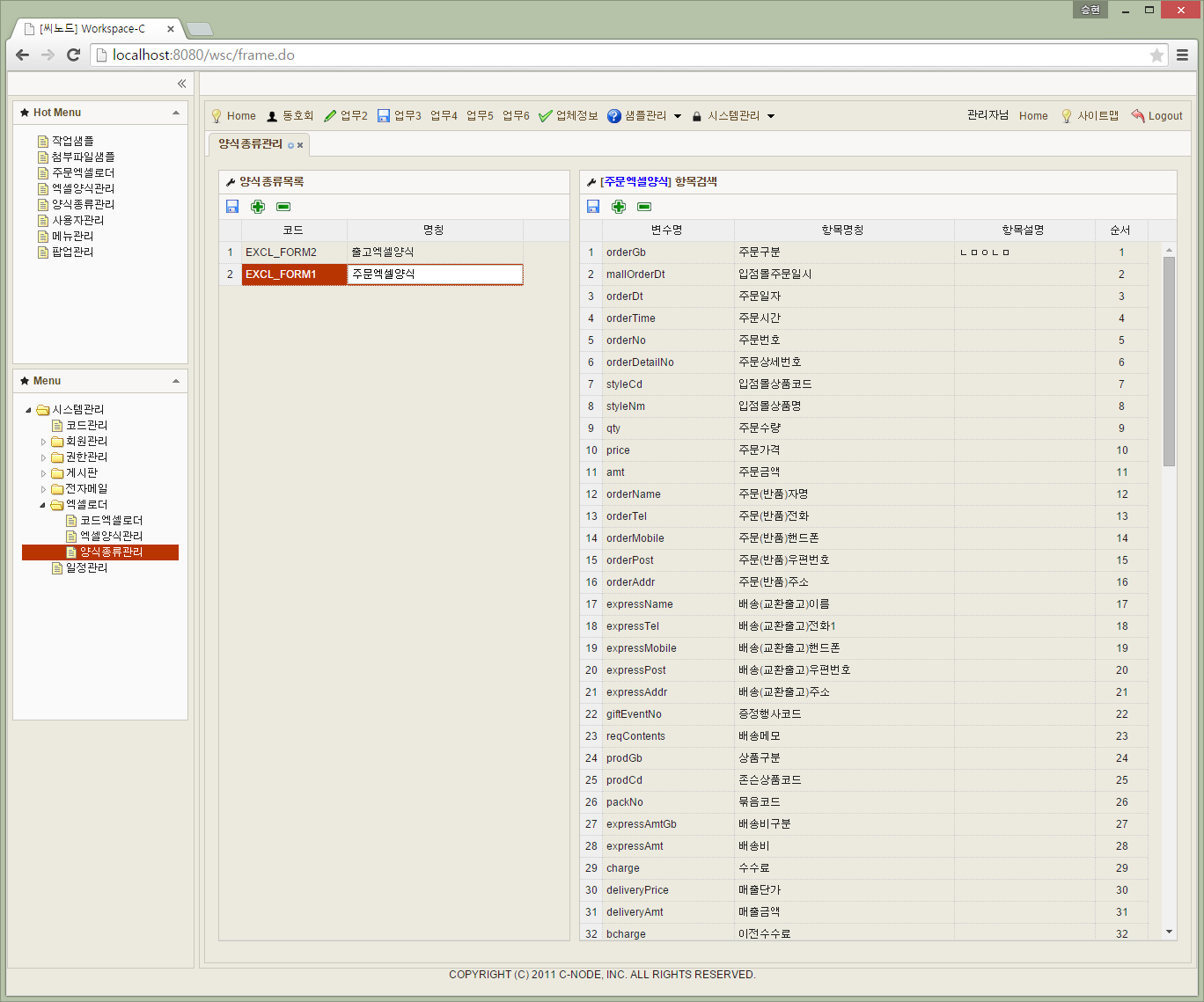
## Excel Loader 맵핑 기능

* 위에서 설명한 엑셀로더의 확장판으로 엑셀양식을 설정하는 관리 기능이 별도로 존재하며, 엑셀파일을 업로드시에 설정되어 있는 맵핑정보에 따라 엑셀을 읽어들이는 기능이다.

1. 양식 종류 관리

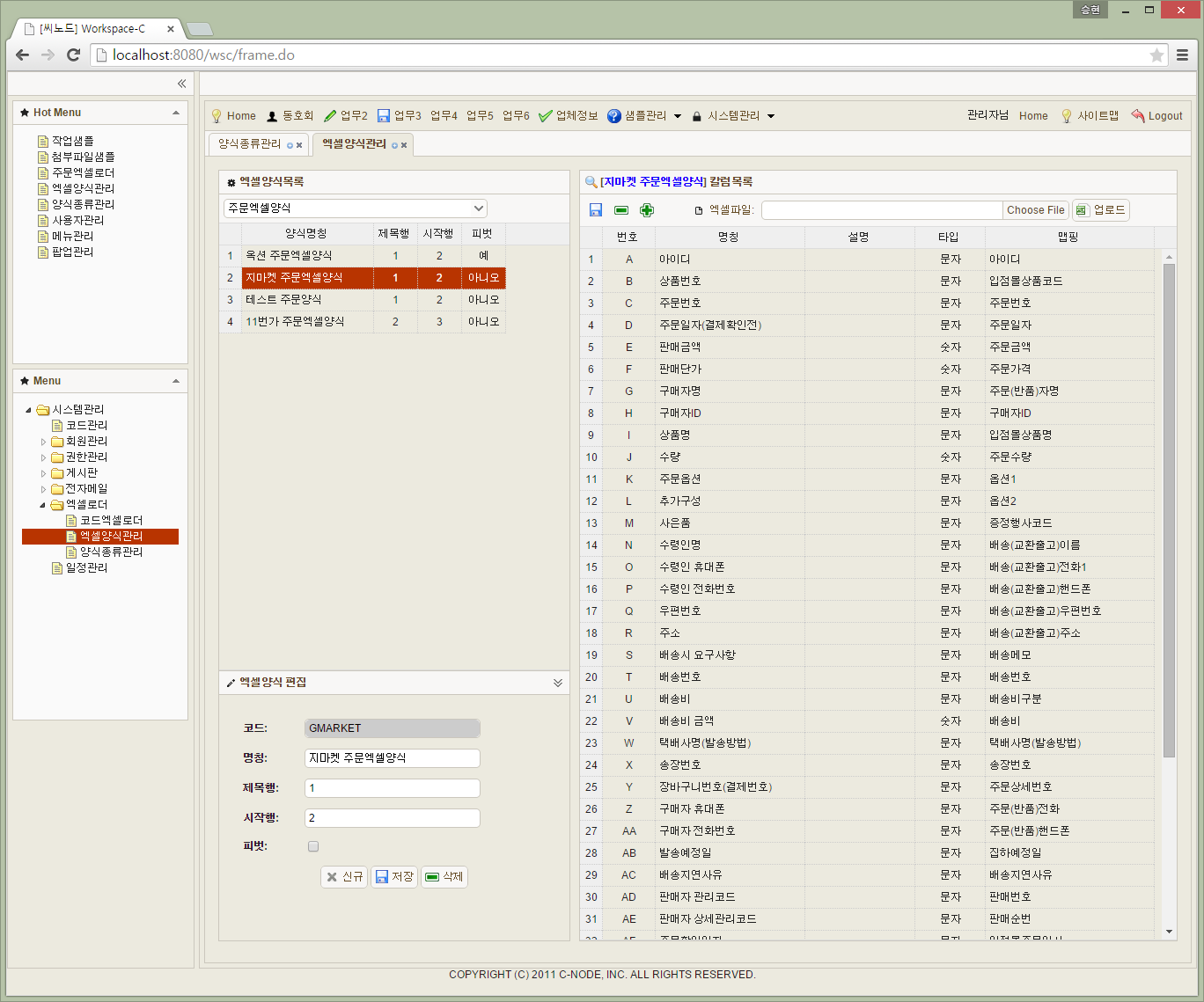
주문 또는 출고 등의 양식의 그룹과 맵핑할 변수명을 정의하는 부분으로 해당 맵핑변수를 대상 테이블의 칼럼에 대입하여 저장처리 기능을 개발자가 구현할 수 있도록 기본 구조를 설정한다.

해당 내용은 코드테이블에 저장되며, 별도의 관리페이지를 만들어 설정이 용이하게 하였다.



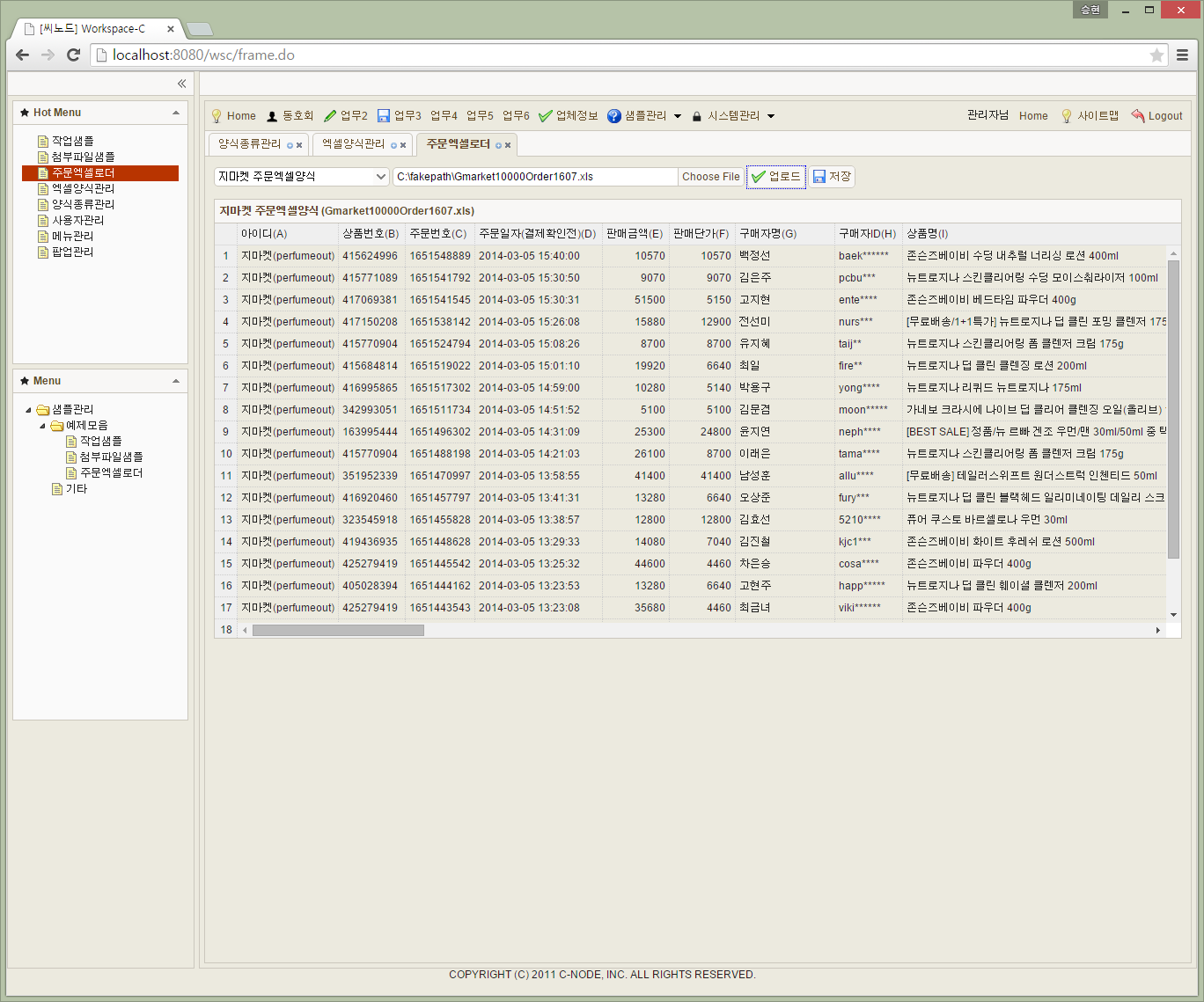
1. 엑셀 양식 관리

각 엑셀의 양식그룹에 해당하는 종류(오픈마켓)별 엑셀양식을 등록하고 개별 칼럼을 설정한다. 이 부분에서 1)번에서 정의한 맵핑변수와 개별 칼럼을 맵핑하도록 한다.



1. 주문 엑셀 로더 (예제)

엑셀양식의 샘플로서 양식에 맞는 엑셀파일을 업로드하고 해당 데이터를 읽어 화면에 그리드형태로 display 하는 기능까지 개발되어 있으며, 저장버튼 클릭 후에 실제 저장을 처리하는 로직은 업무적인 요청에 따라 별도의 개발이 필요하다.



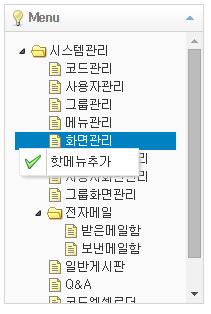
## EasyUI Grid 기능

* 화면의 목록은 모두 EasyUI의 Grid 를 사용하여 구현되어 있으며, 해당 Grid 의 Control 을 위해서 business.js 에 jeasygrid 라는 컴포넌트를 만들어 사용하고 있다. jeasygrid 는 그리드의 검색 외에도 그리드의 편집, 행 삭제, 저장 등의 처리를 수행하는 함수들이 구현되어 있다.
* 각 관리 기능별로 사용된 그리드의 유형은 다음과 같으며, 해당 기능이 필요할 경우 각 예제를 참고하여 구현하도록 한다.

|  |  |  |  |
| --- | --- | --- | --- |
| 그리드 유형 | 메뉴 | 구현소스 | 기능 특징 |
| 팝업폼 그리드 | 사용자관리 | user.js | 기본적인 페이징 그리드  행선택시 상세내용 레이어 팝업 보기  추가 시 팝업 폼으로 입력 |
| 트리형 그리드 | Q&A | reply.js | 트리 형태의 페이징 그리드  상세 및 등록은 팝업 폼 그리드와 동일 |
| 폼패널 그리드 | 코드관리 | code.js | 기본적인 페이징 그리드  행선택시 상세내용 하단 폼 패널 표시  하단 폼 패널 등록 및 삭제 처리  검색 시 정렬옵션 선택 |
| 행편집 그리드 | 메뉴관리 | menu.js | 기본적인 페이징 그리드  행선택시 행 단위 편집 기능 |
| 다중 그리드 | 사용자그룹관리 | usergroup.js | 3개의 그리드 동시 표시  그리드 행 편집 기능  2개 그리드 선택 행을 하단 그리드에 행 추가 |

* 기본적으로 그리드는 페이징을 기본으로 하며 페이징이 필요없는 경우 jeasygrid 의 init 에 넘길 Arguments 에서 “pagination:**false**”로 설정한다.
* 그리드의 정렬은 Local 정렬을 기본으로 하며, 별도의 서버정렬이 필요한 경우 “코드관리”에 구현된 것과 같이 검색 옵션으로 처리하도록 한다. 정렬할 칼럼은 다음과 같이 해당 항목의 data-options 에 “sortable:**true**”를 추가하도록 한다.

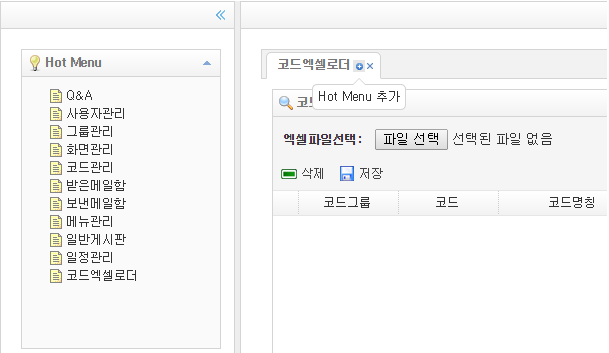
## Context Menu 기능

* 좌측메뉴에 현재 EasyUI 의 Context Menu 기능이 구현되어 있다. 좌측메뉴에서 오른쪽마우스를 클릭하면 다음의 그림과 같이 “핫메뉴추가” 라는 Context Menu가 보인다.

|  |
| --- |
| this.tree.tree({  …,  onContextMenu: function(e,node) {  e.preventDefault();  $(this).tree('select',node.target);  p.mctx.menu('show',{left: e.pageX,top: e.pageY});  }  }); |

* 해당 항목에 대해 별도의 기능버튼이 필요한 경우에 사용할 수 있는 기능으로서 해당 부분을 참고하기 위해서 widget.js 의 submenu 또는 hotmenu 를 확인하면 된다. 이는 EasyUI 에 있는 기능으로 EasyUI의 API 를 참고하여 구현하도록 하며, submenu 상에 구현된 예시는 다음과 같다.

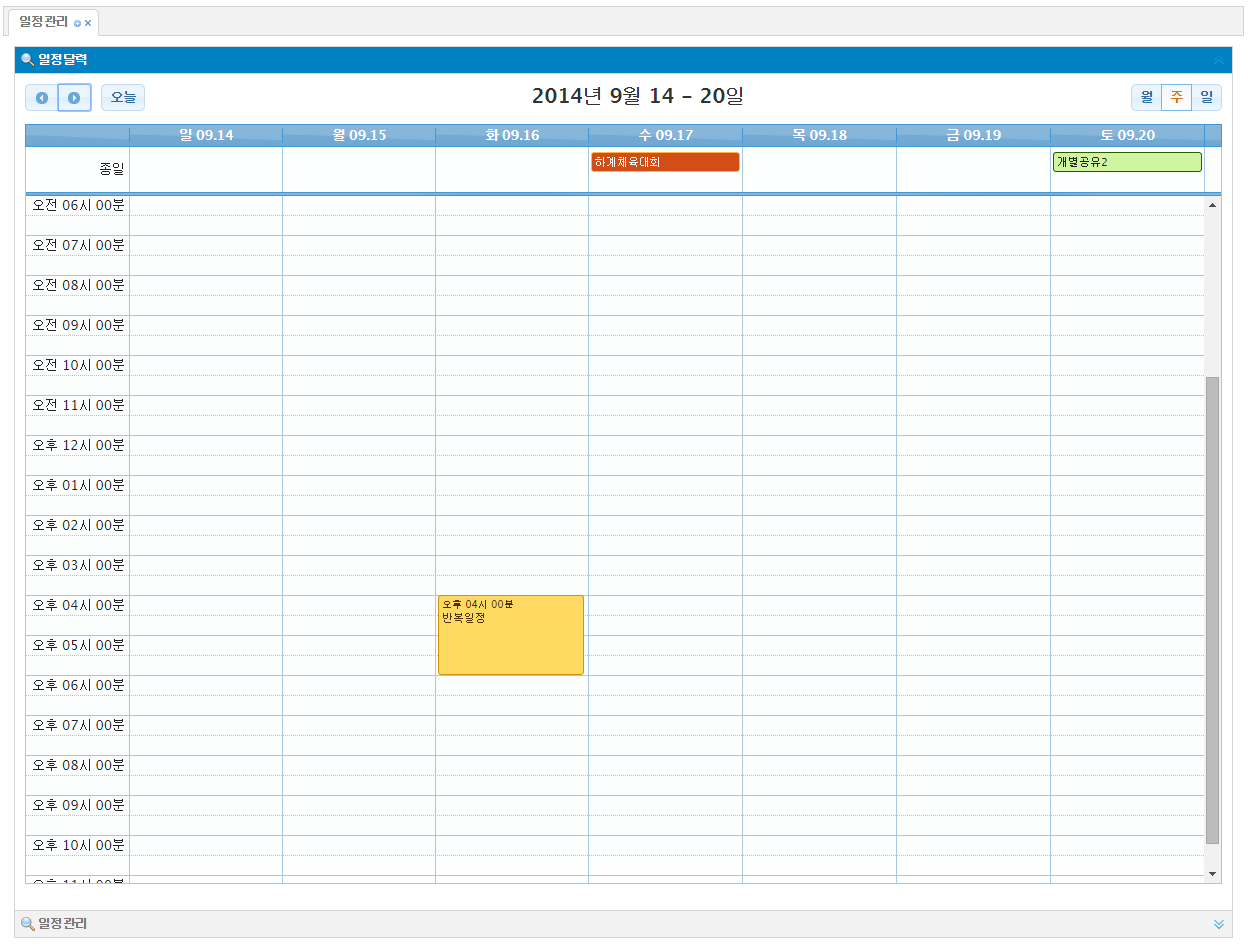
## Hot Menu 기능

* Hot Menu 는 사용자가 자주 찾는 메뉴를 Hot Menu에 등록하고 해당 메뉴의 접근을 편하게 하기 위해 구현된 기능이다. Hot Menu 의 추가는 열린 탭 패널의 아이콘이나, 좌측 메뉴의 Context Menu를 이용해 추가할 수 있다. 이는 widget.js 안의 hotmenu 객체에 정의되어 있으며, 탭 패널과 Layout 의 West Region 을 이용하는 경우 별도의 설정 없이 기본적으로 사용된다.

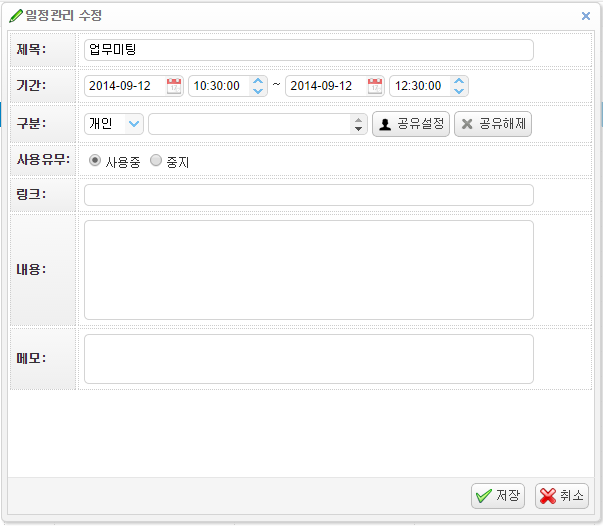
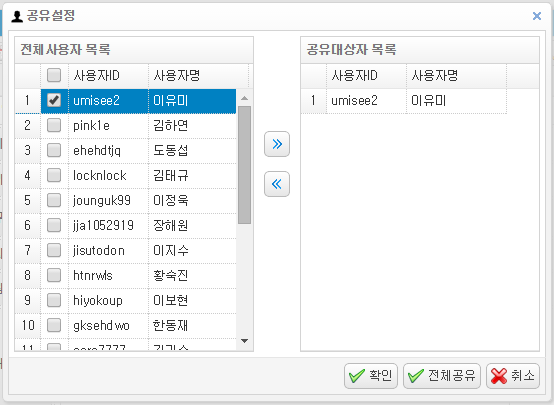
## 일정관리 기능

* 일정관리는 Full Calendar jQuery Plugin을 사용하여 개발되었으며, 개인의 일정은 Grid를 사용하여 관리되며, Full Calendar 는 월별, 주별, 일별로 외부에서 공유한 일정까지 함께 조회되는 기능이다.
* 일정달력의 표시예제는 다음과 같다.





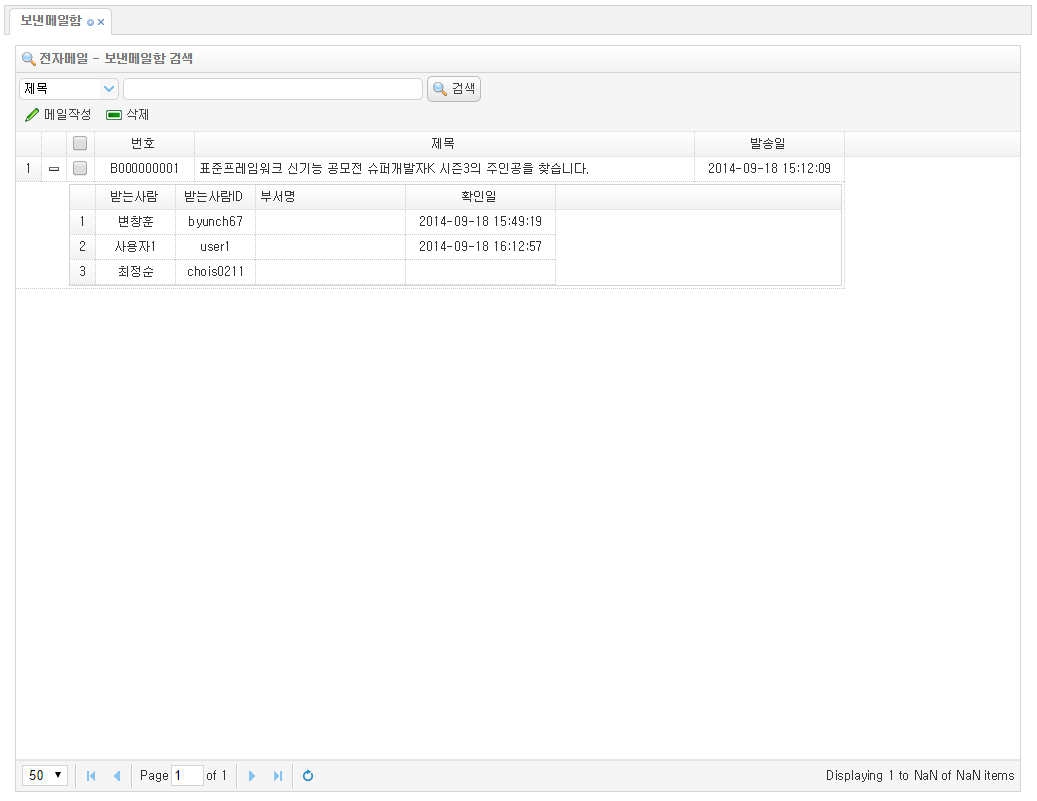
* 공유일정을 등록하기 위해서 공유 설정을 해야 하며, 그림과 같이 공유설정 버튼을 클릭하면 전체사용자에서 선택할 수 있는 레이어팝업이 띄워진다. 공유할 대상을 선택 후 확인버튼을 누르면 선택한 대상자가 표시된다. 만약 전체 사용자를 대상으로 공유할 경우엔 공유설정 팝업에서 하단의 “전체공유” 버튼을 누르면 된다.

* 추가적으로 일정관리 기능이 필요할 경우엔 게시판을 추가하는 방법과 동일한 패턴으로 구현하면 된다.

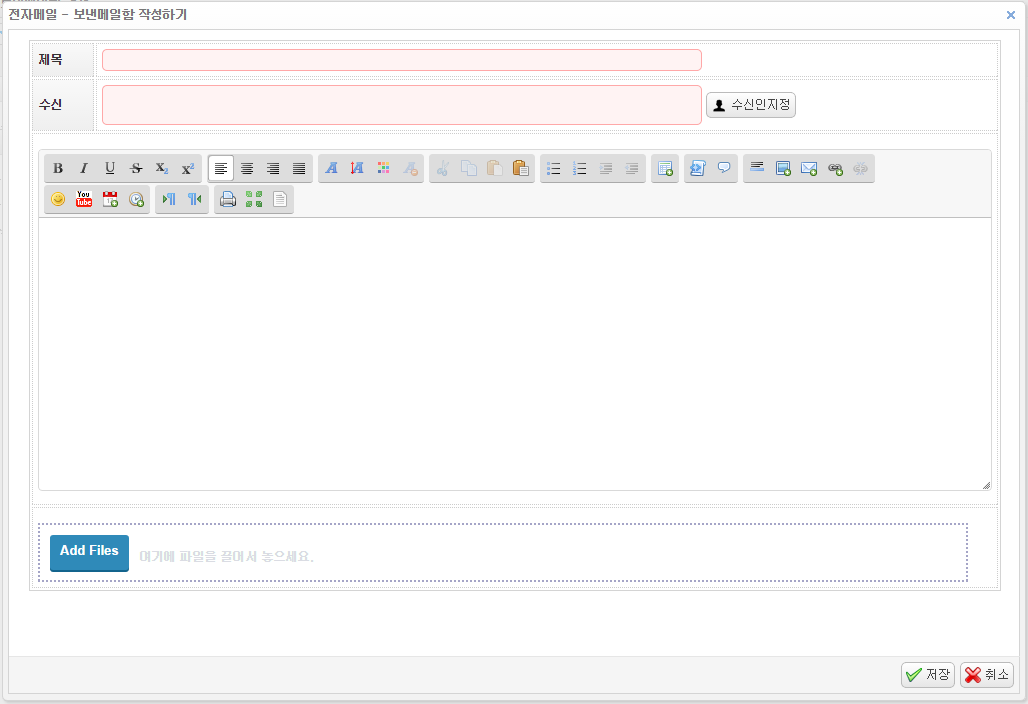
## 전자메일 기능

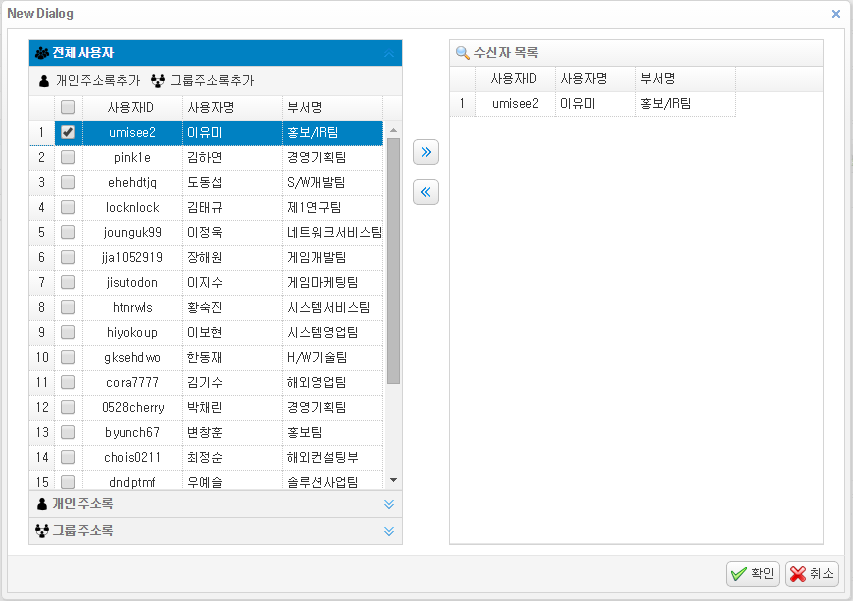
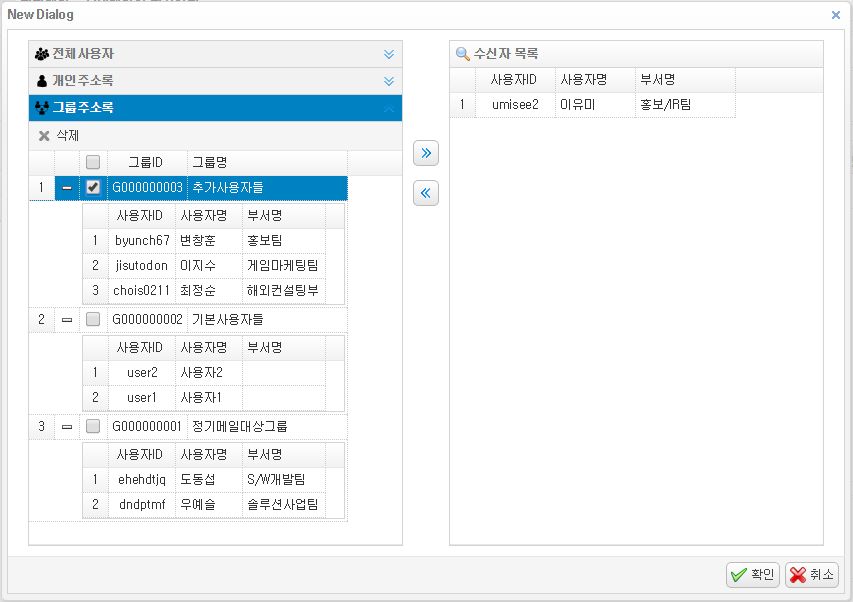
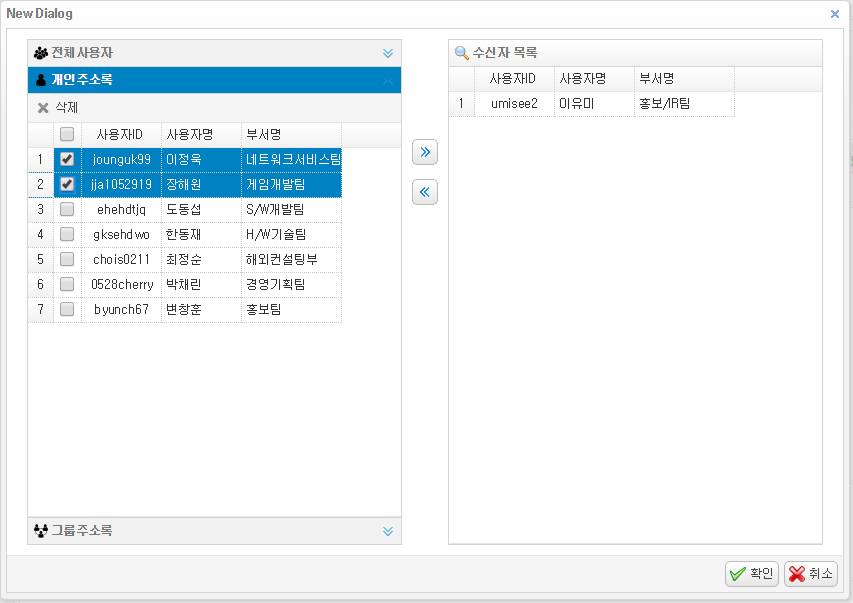
* 게시판을 활용한 기능으로 게시판 + 게시대상 설정의 기능을 가지고 있다. 다만, 받은메일함과 보낸메일함의 검색기준이 다르므로 별도의 URL로 맵핑되어 구현되어 있다. 메일을 삭제시에도 받은 대상자는 해당 메일을 확인할 수 있어야 하므로 게시판과 다르게 삭제시엔 사용유무 플래그의 비활성화로 삭제처리가 실행된다.
* 보낸메일함의 경우 대상자가 다수이므로 그림과 같이 서브그리드를 활용한 방식으로 구현되어 있다.



* 게시대상 설정은 다음과 같이 전제사용자에서 선택, 개인주소록에서 선택, 그룹 주소록에서 선택의 3가지 선택방식을 가지고 있다.

수신인지정 클릭





## iReport 연동 기능

* iReport 연동은 현재 기본적인 개발만 이루어진 단계이므로, 추가적인 개발이 필요한 부분이다.
* **현재까지 개발된 기능은 다음과 같다.**

1. iReport Libaray Import
2. Spring Framework 내의 Report 설정 파일 생성(wsc-jasper.xml)
3. Spring Framework 의 JasperReportsMultiFormatView 를 사용한 PDF, HTML, XLS Viewer 설정
4. 코드데이터 검색 샘플을 활용한 iReport 파일 생성
5. 코드관리 내에 PDF, XLS, HTML 형태의 팝업보기 설정

* **개발이 필요한 부분**

1. HTML 방식의 리포트 이미지 표시 방법
2. 출력 실행 처리 부분

* **Report 구현 절차**

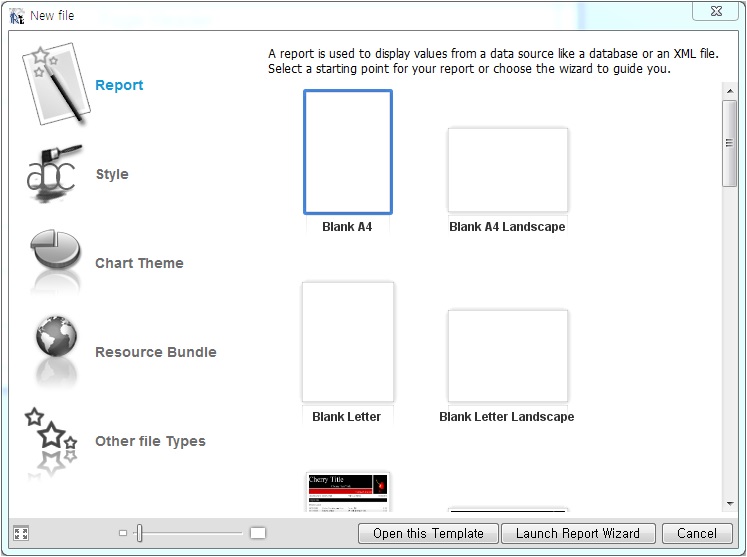
|  |
| --- |
| 1. 리포트 파일을 생성하여 {웹경로}/WEB-INF/reports 안에 위치시킨다. |
| **C:\Users\carmine\Desktop\가이드문서작업\리포트경로.png** |

|  |
| --- |
| 1. wsc-jasper.xml 에 아래와 같은 구문을 추가한다. 단, id 는 고유한 아이디로 정의되어야 하며 여기에 정의한 id가 controller 의 view name으로 사용된다. url 에는 생성한 리포트 파일을 입력한다 |
| <bean id=*"codeReport"* class=*"org.springframework.web.servlet.view.jasperreports.JasperReportsMultiFormatView"*>  <property name=*"url"* value=*"/WEB-INF/reports/report5.jasper"*/>  <property name=*"exporterParameters"*>  <map>  <entry key=*"net.sf.jasperreports.engine.JRExporterParameter.CHARACTER\_ENCODING"*>  <value>UTF-8</value>  </entry>  <entry key=*"net.sf.jasperreports.engine.export.JRHtmlExporterParameter.IS\_USING\_IMAGES\_TO\_ALIGN"*>  <value>false</value>  </entry>  </map>  </property>  </bean> |

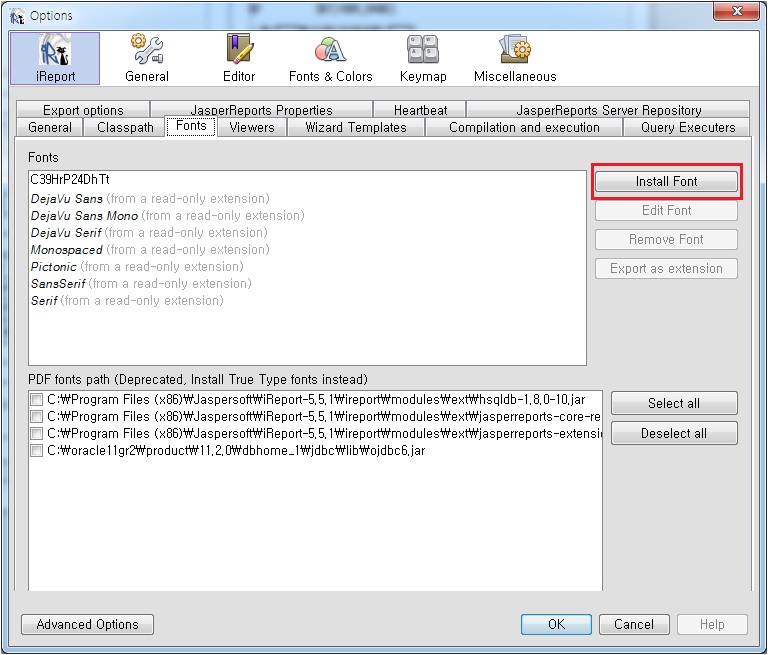
|  |
| --- |
| 1. 해당 업무의 Controller Class에 다음과 같이 리포트 관련 메서드를 추가한다. |
| @RequestMapping("/code.{format}")   |  | | --- | | 1. 해당 업무의 JSP 파일에는 해당 버튼을 추가하고 스크립트에 해당 버튼의 이벤트를 다음과 같이 추가한다. | | $(**function**() {  $("#report-button-pdf").bind("click", doReport);  $("#report-button-xls").bind("click", doReport);  $("#report-button-htm").bind("click", doReport);  });  **function** doReport() {  **var** id = **this**.id;  **var** ext = 'pdf';  **if** (id == 'report-button-pdf') ext = 'pdf';  **else** **if** (id == 'report-button-xls') ext = 'xls';  **else** **if** (id == 'report-button-htm') ext = 'html';  jlogic.report({url: getUrl("/common/report/code.")+ext});  } |   **public** String reportCode(HttpServletRequest request, Model model, @PathVariable("format") String format) **throws** Exception {  ParamsMap params = getParams(request);  Object data = service.search(params);  JRDataSource JRdataSource = **new** JRBeanCollectionDataSource((List) data);  model.addAttribute("datasource", JRdataSource);  model.addAttribute("format", format);  **return** "codeReport";  } |

## iReport Jaspersoft연동

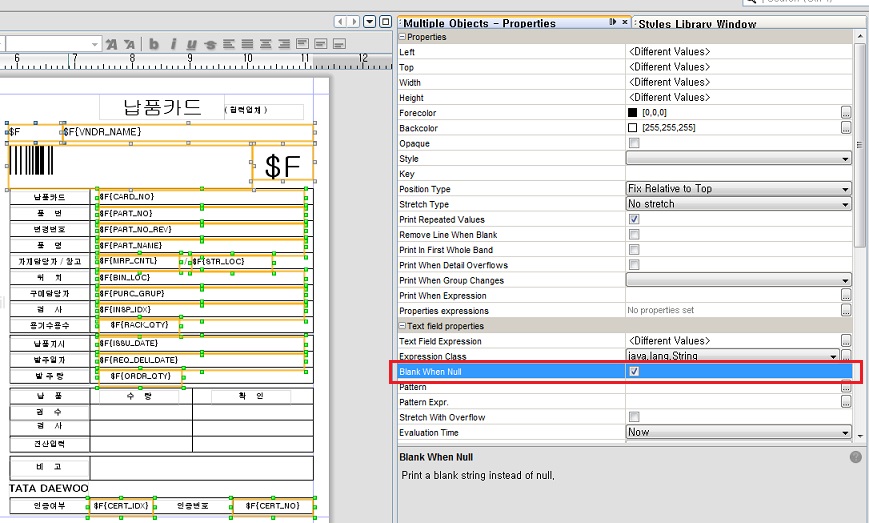
* Jaspersoft server iReport 작성방법
  + iReport 생성



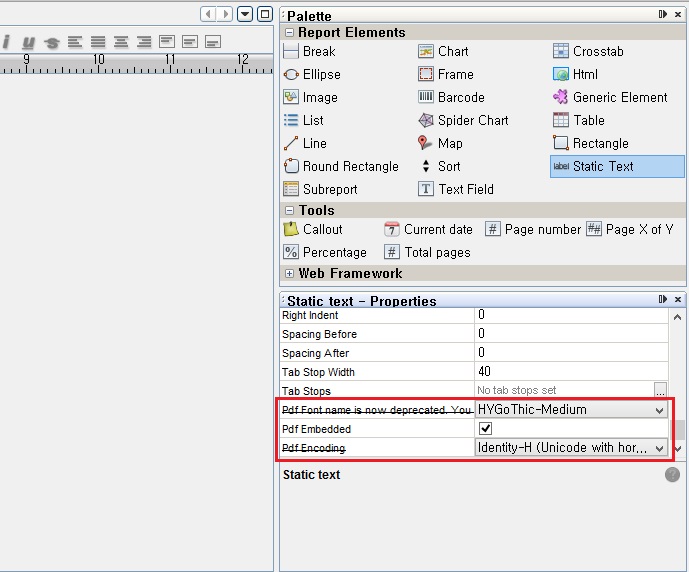
* + 폰트 추가



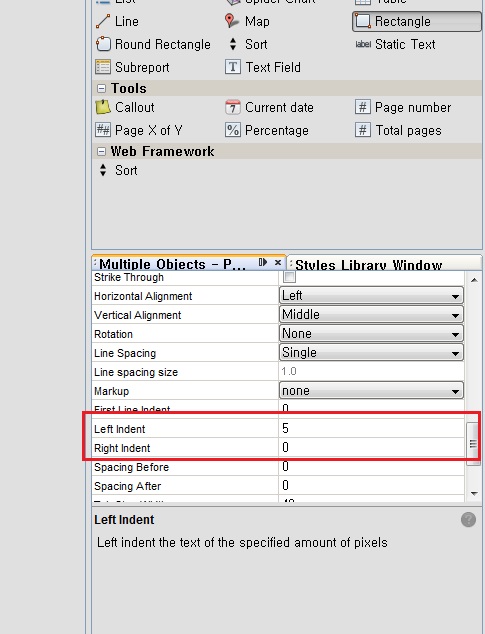
* + 널값처리



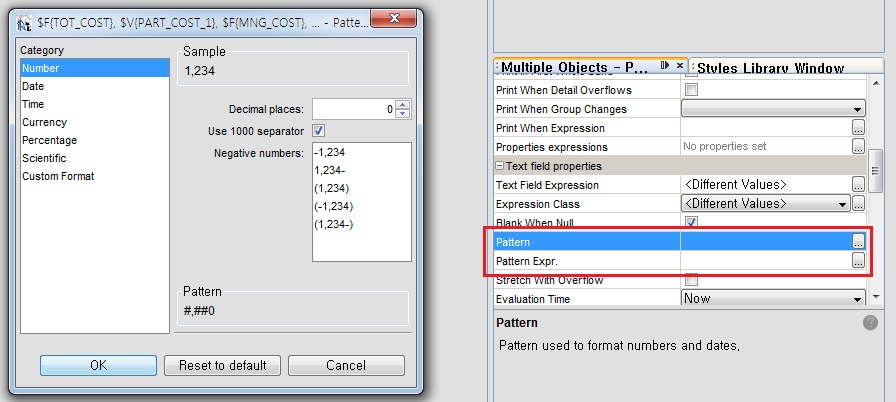
* + pdf저장시 한글 인코딩 처리



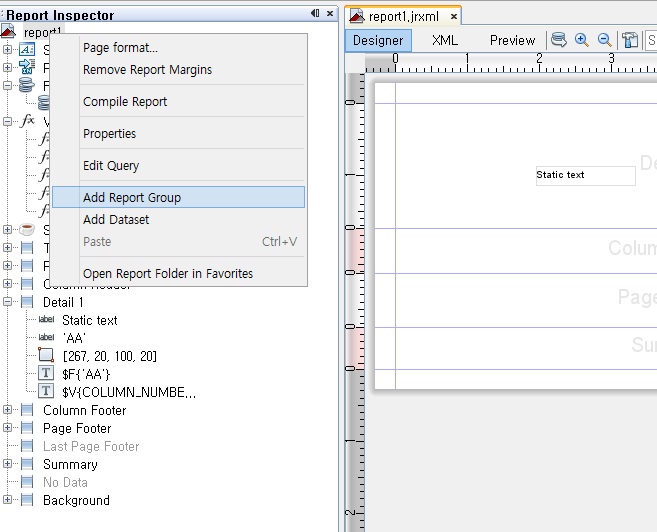
* + 텍스트 들여쓰기



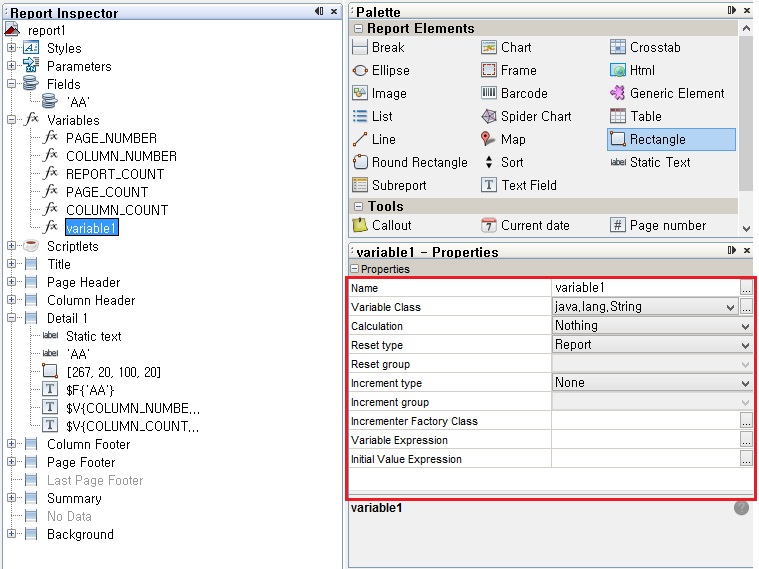
* + 숫자포멧



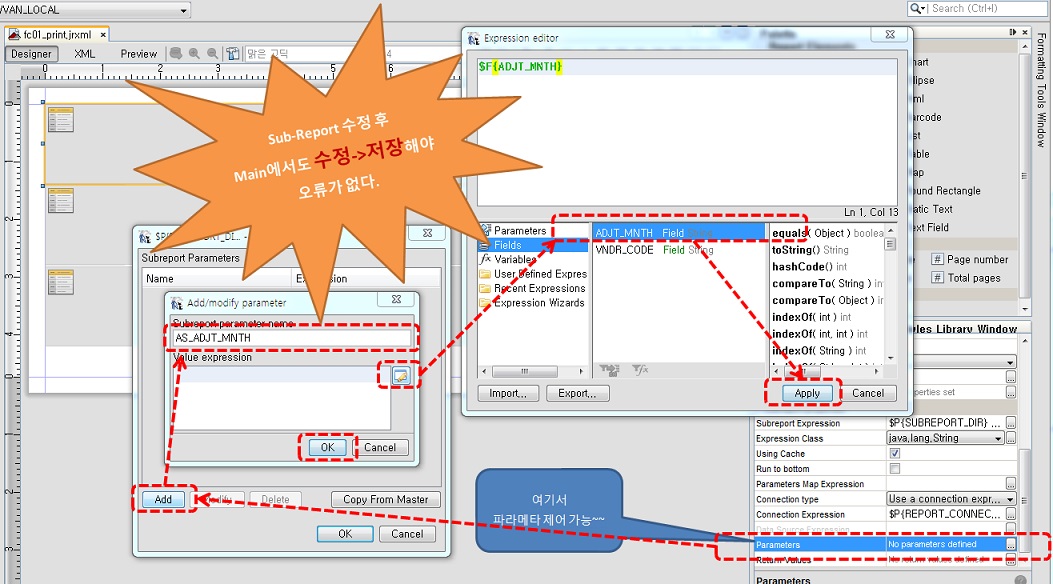
* + 그룹등록



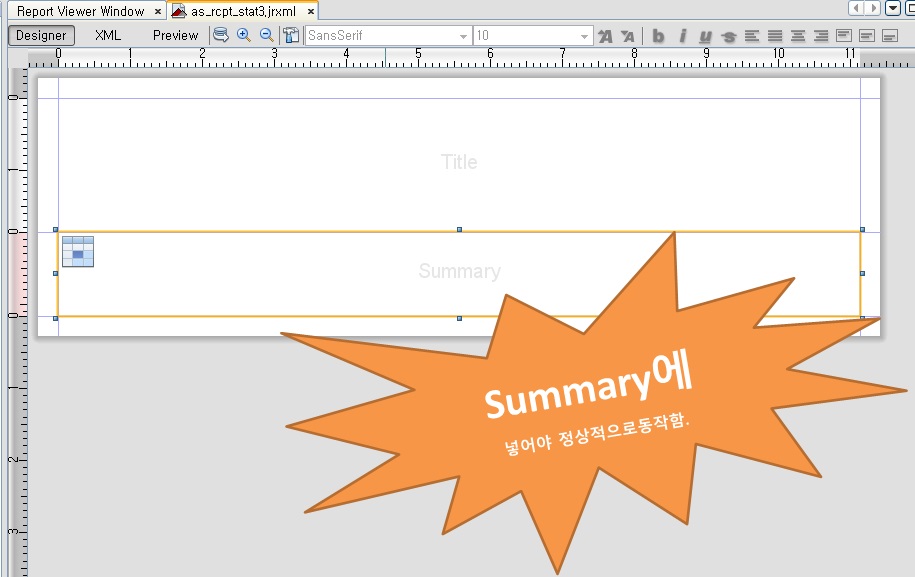
* + 변수 계산식



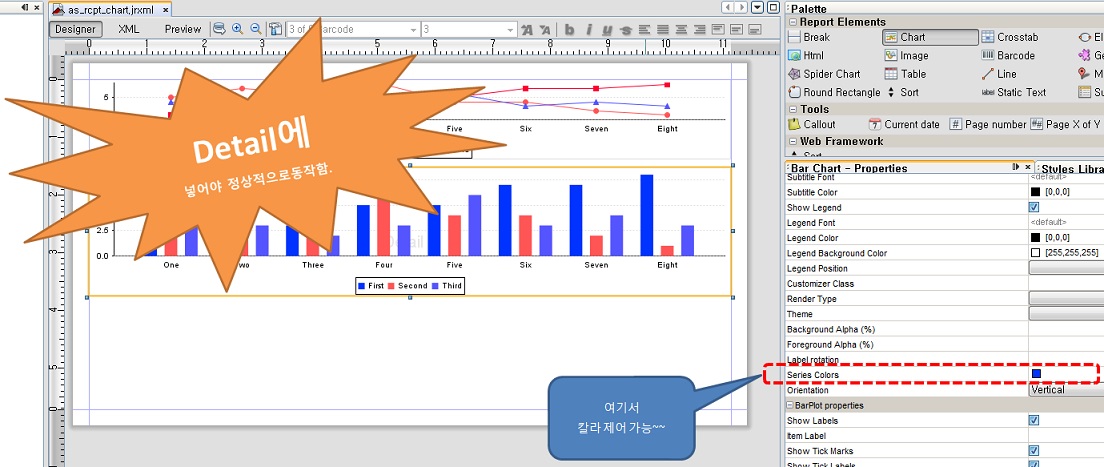
* + 서브 레포트



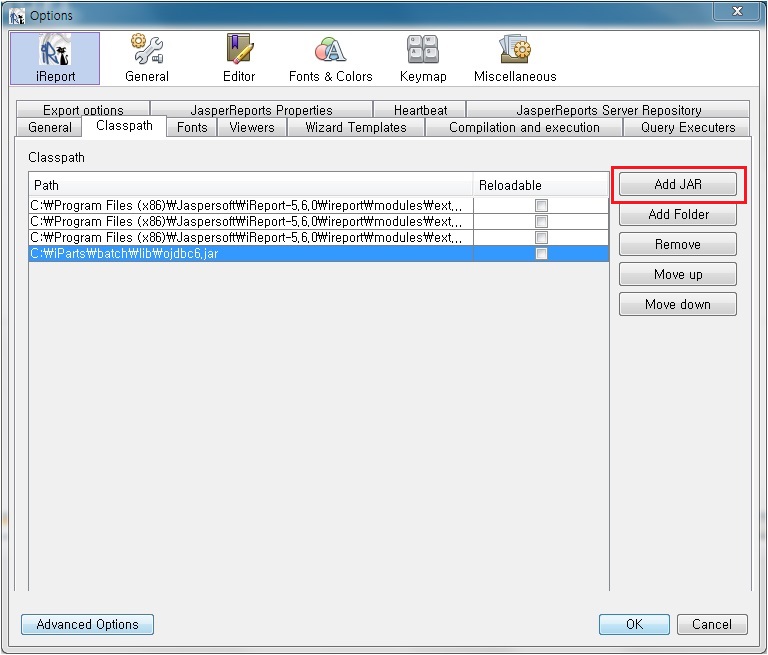
* + 크로스탭



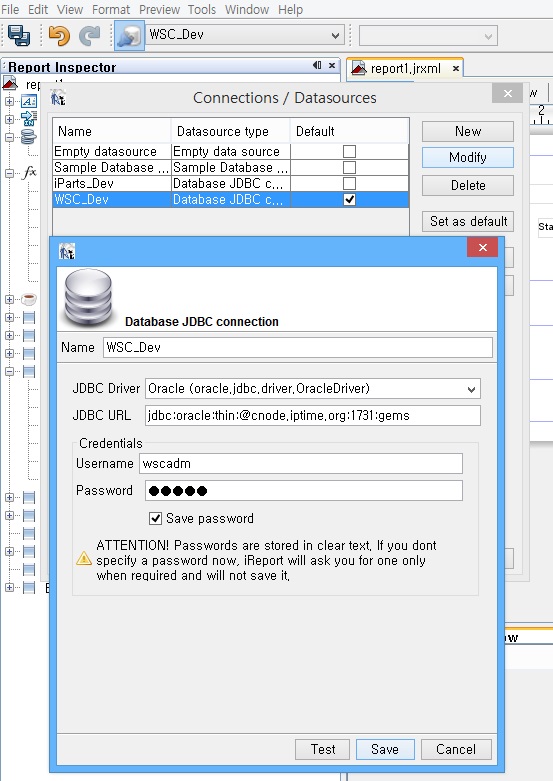
* + 차트



* + Ojdbc 추가



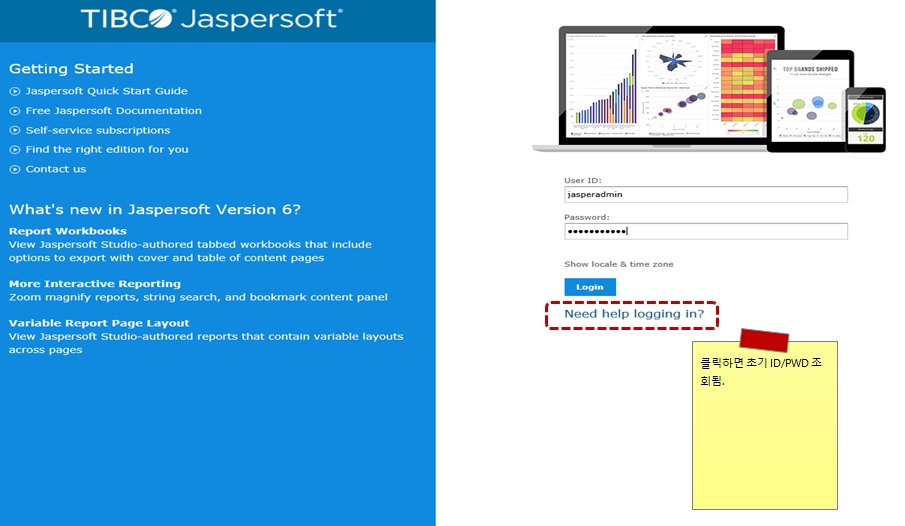
* + DB연결



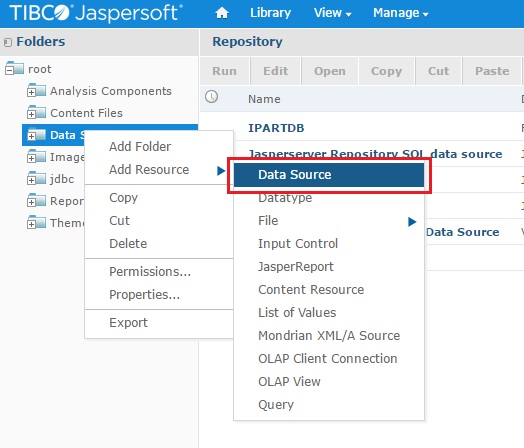
* + 툴바 수정 – JasperSoft 설치경로 \apache-tomcat\webapps\jasperserver\WEB-INF\jsp\modules\viewReport\ViewReport 수정

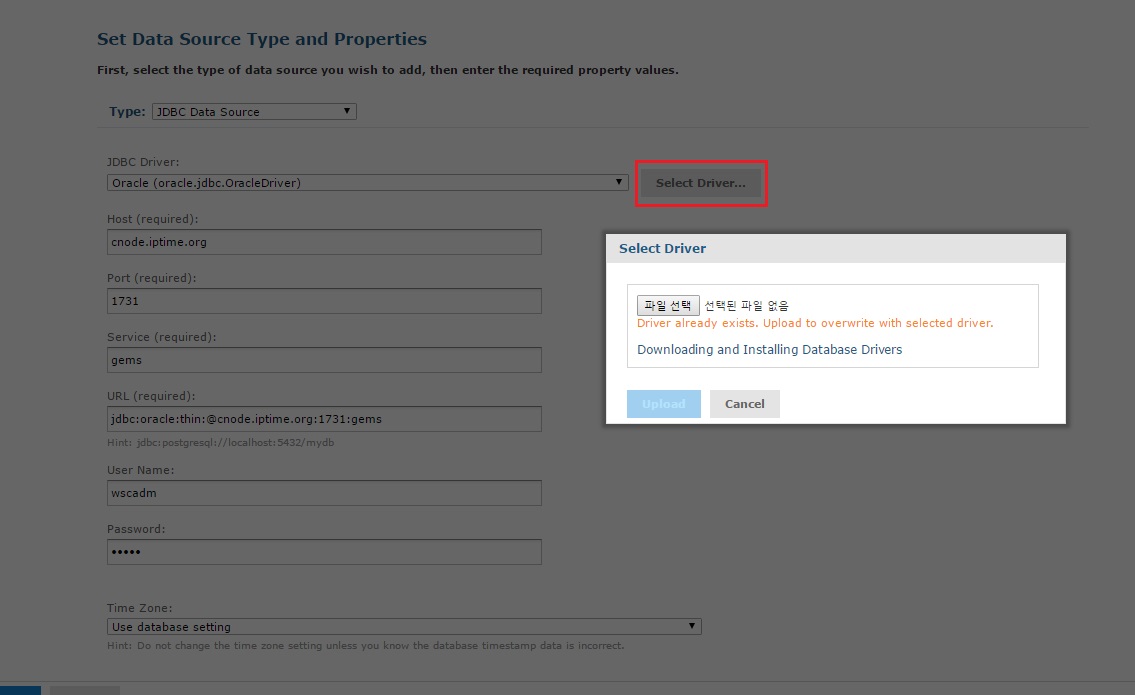


* + Jaspersoft 로그인

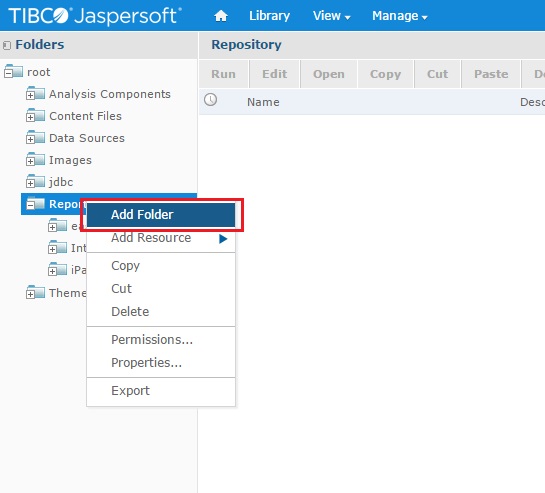


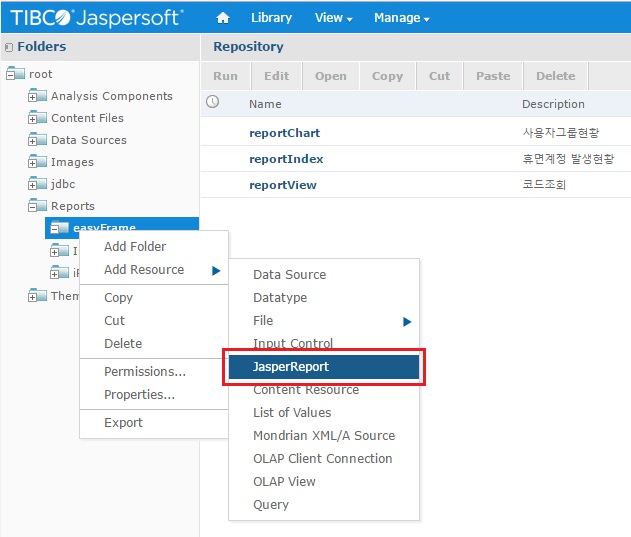
* + Jaspersoft DataSource(DB연결)추가



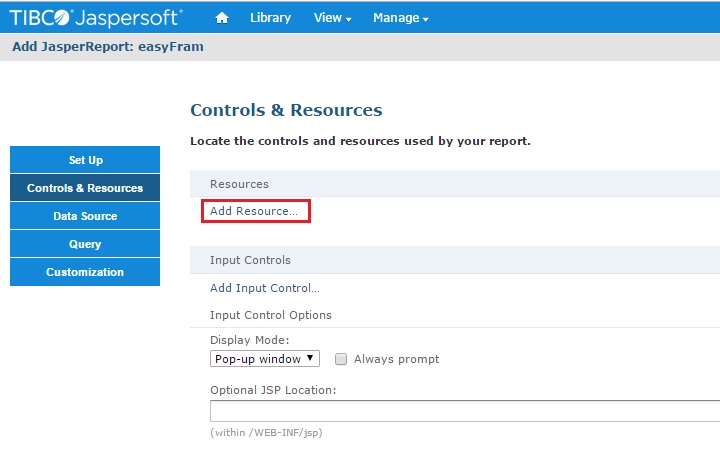


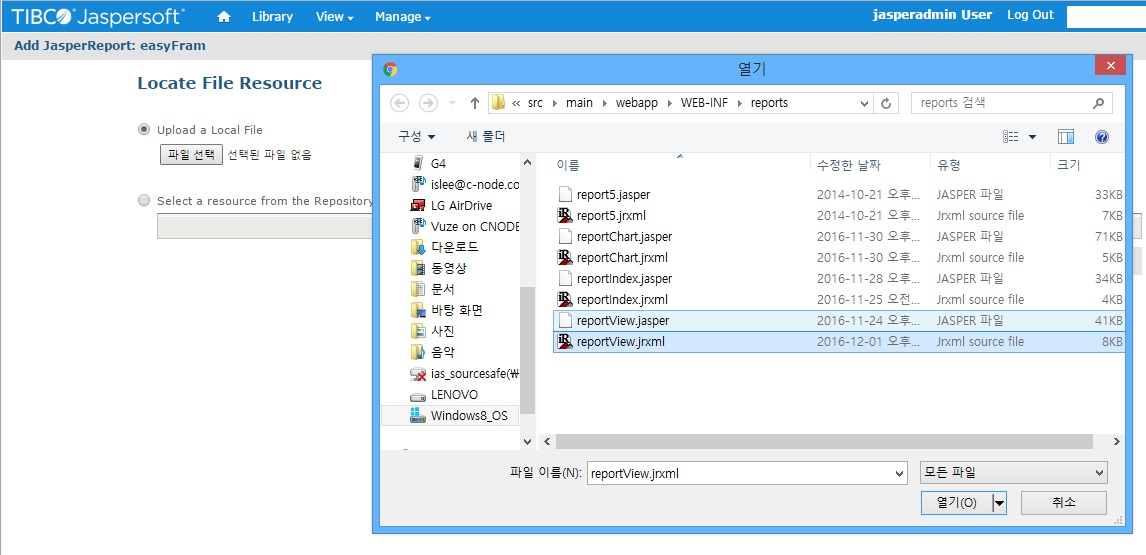
* + Jaspersoft Report 추가

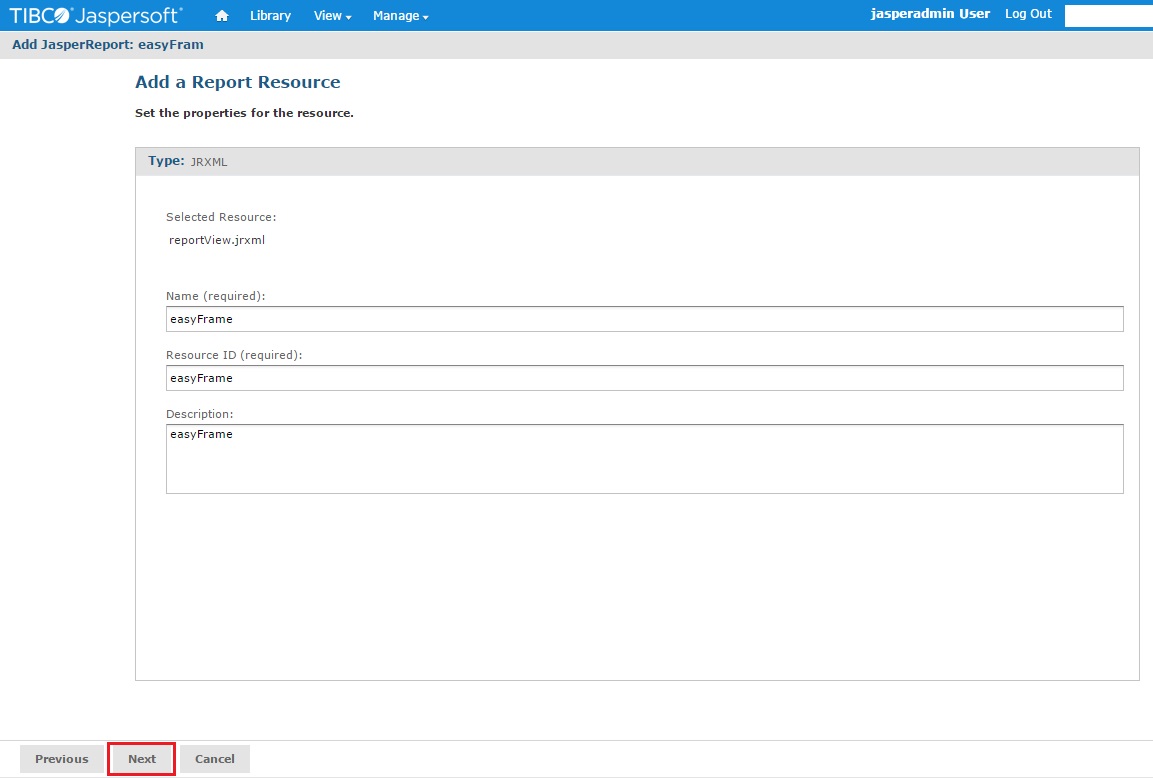


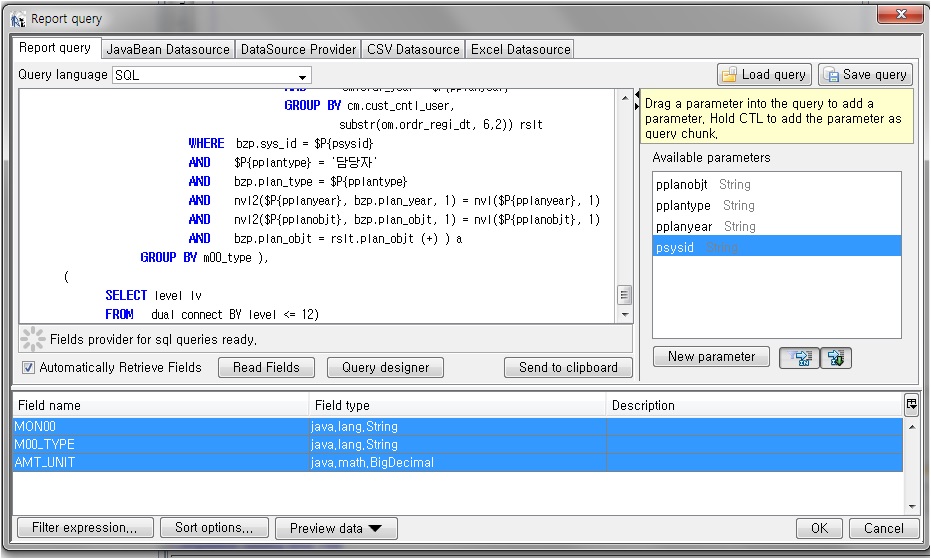


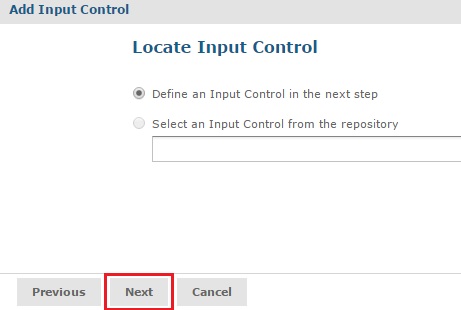


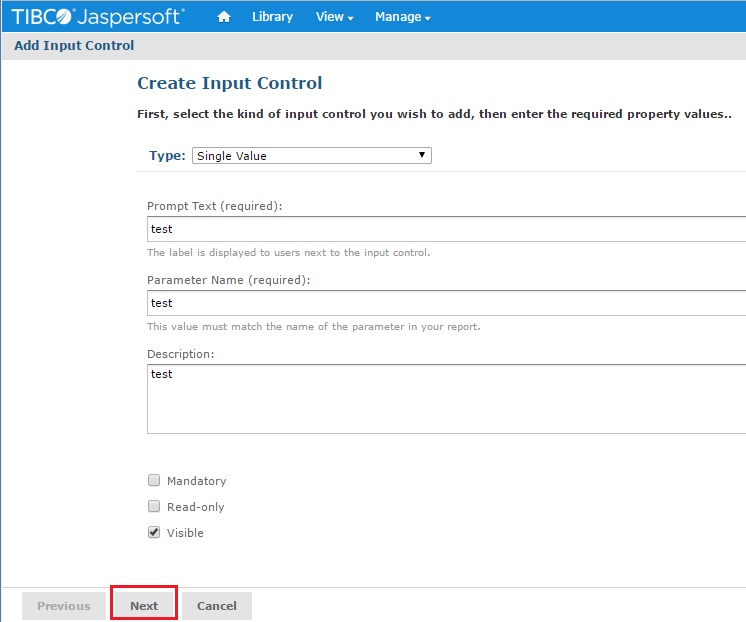


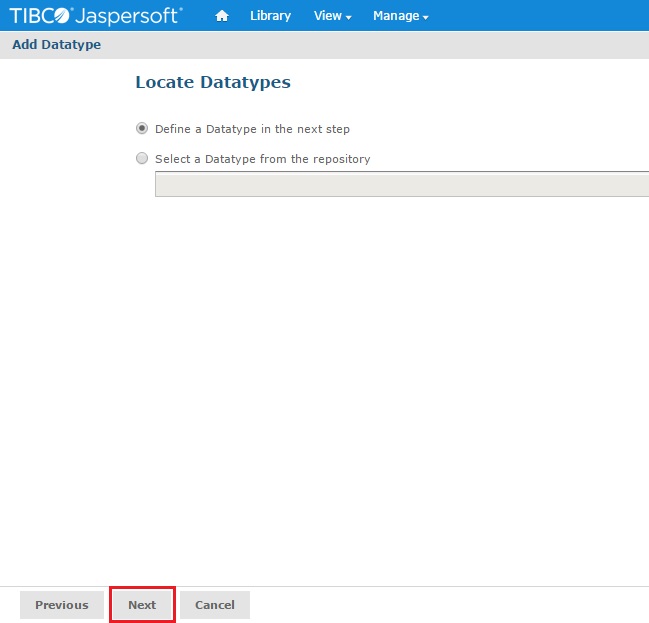


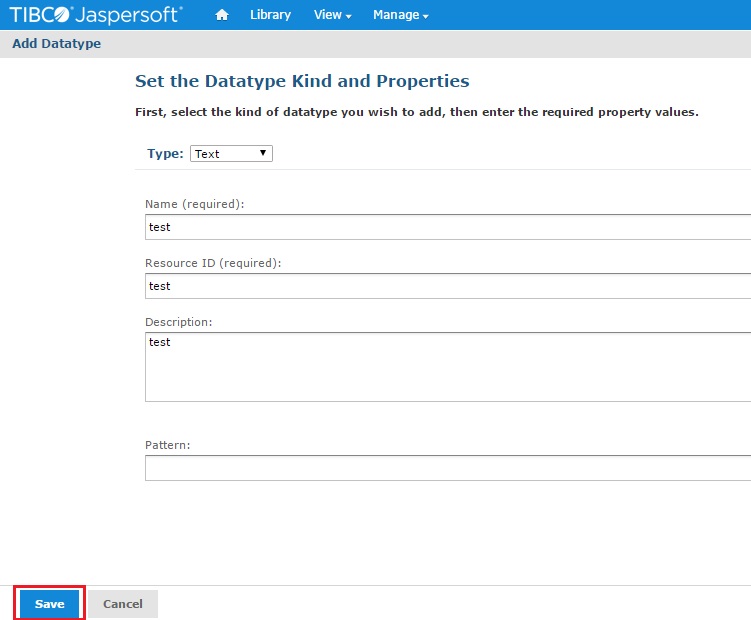


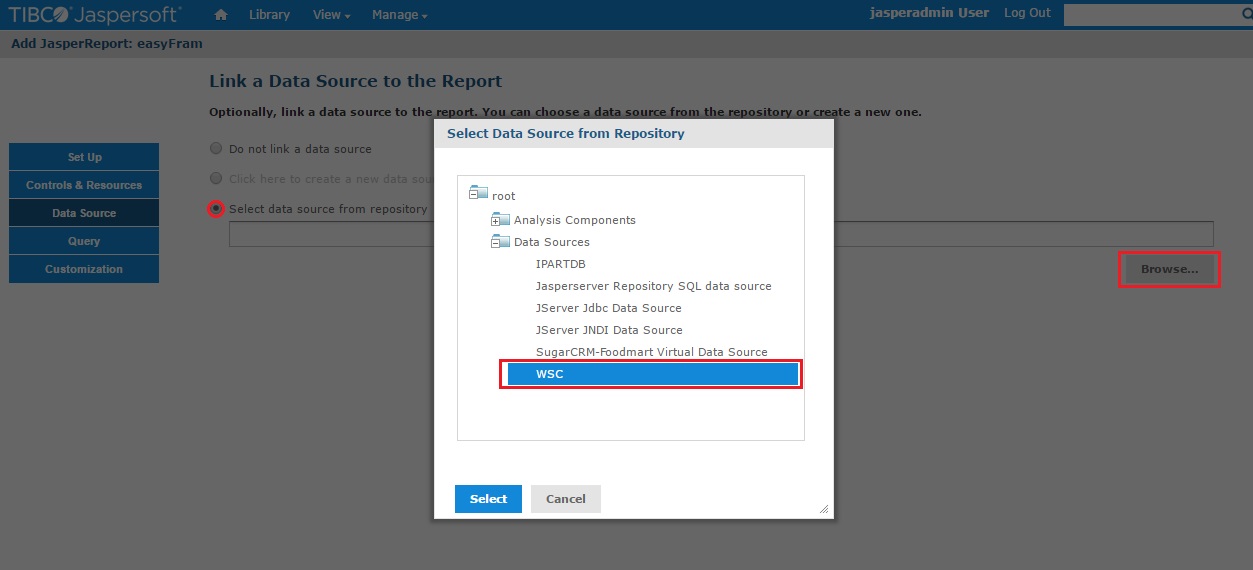




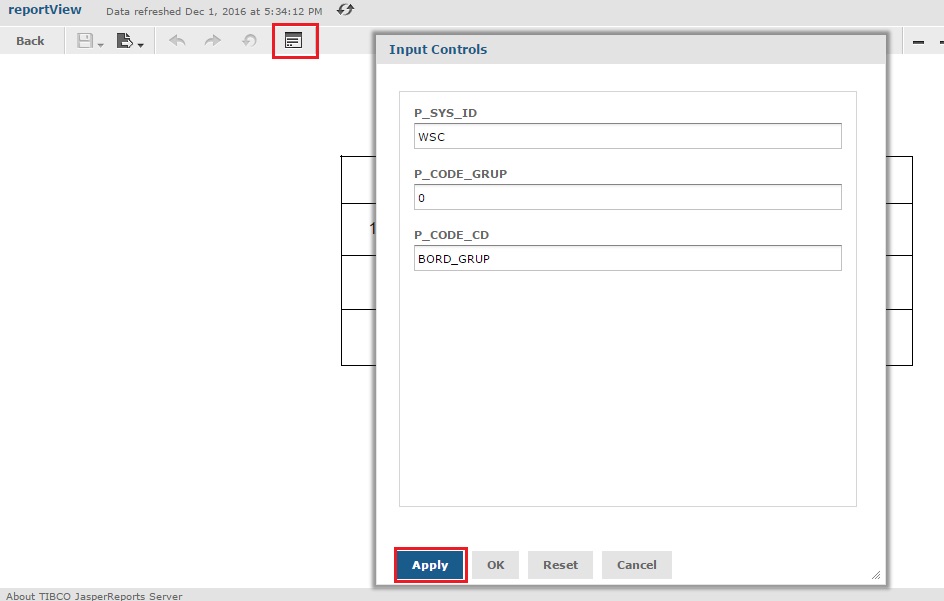








* + Jaspersoft Report 확인



## EasyUI Theme 변경

* EasyUI 에서 제공하는 테마는 현재 총 5가지이며 default/black/bootstrap/gray/metro 가 있다. 이는 easyui-1.4/themes 안에서 확인할 수 있으며, 현재는 bootstrap 이 적용되어 있다.
* 해당 테마를 변경하기 위해서 다음과 같이 수정한다.

1. src\main\webapp\WEB-INF\views\include\common.jsp 파일을 편집기에서 오픈한다.
2. 다음의 코드에서 노락색으로 칠해진 부분은 위 작성한 keyword중 하나로 변경하면 해당 테마가 적용된다.

<link rel=*"stylesheet"* type=*"text/css"* href=*"*<c:url value=*"/resources/jquery/easyui-1.4/themes/****bootstrap****/easyui.css"* />*"* />

1. 만약 별도의 테마를 생성해서 적용하고자 할 경우엔 <http://www.jeasyui.com/themebuilder/index.php> 에서 원한는 형태의 css 를 만든 후 웹어플리케이션에 import 하여 위와같이 변경하면 된다.

* 다만, 일정관리는 별도의 플러그인을 사용하므로 별도로 테마가 적용되어야 한다. 기본적으로 일정달력은 jquery-ui 의 테마가 적용되므로 <http://jqueryui.com/themeroller/> 에서 원하는 형태의 테마를 다운받아 src\main\webapp\resources\jquery\fullcalendar-2.1.1\lib 폴더안의 redmond 나 cupertino 와 동일한 패턴으로 import 한 후 아래의 절차에 따라 변경하면 된다.
* 일정관리의 달력의 테마를 변경하기 위해서 다음과 같이 수정해야 한다.

1. src\main\webapp\resources\jquery\jquery.schedule.js 파일을 편집기에서 오픈한다.
2. 다음의 구문에서 노란색으로 칠해진 부분은 원하는 테마의 명칭으로 변경하면 된다. 현재는 redmond 가 적용되어 있다.

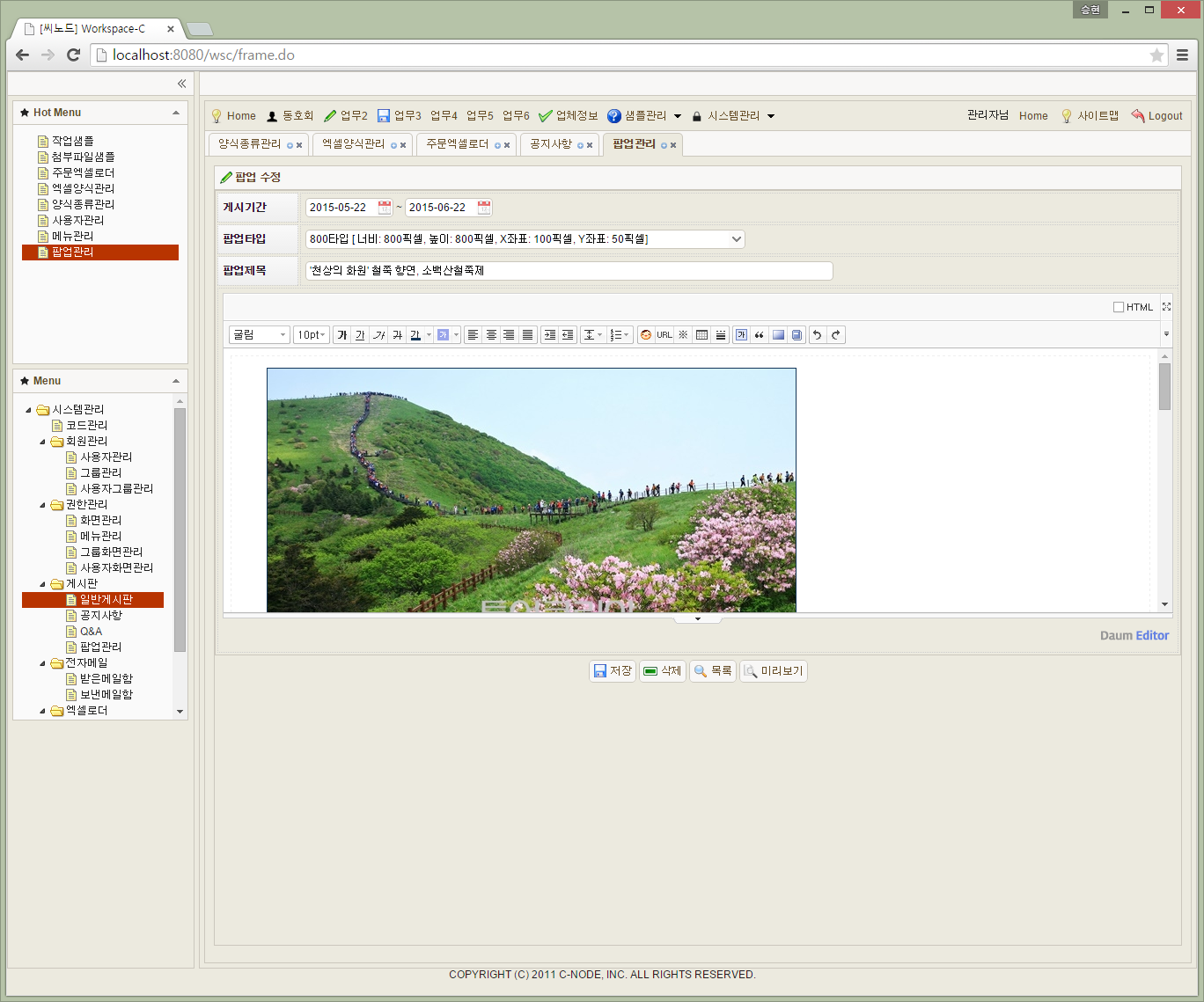
document.write('<link rel="stylesheet" type="text/css" href="'+ jcalendar\_abpath +'/lib/**redmond**/jquery-ui.min.css" />');

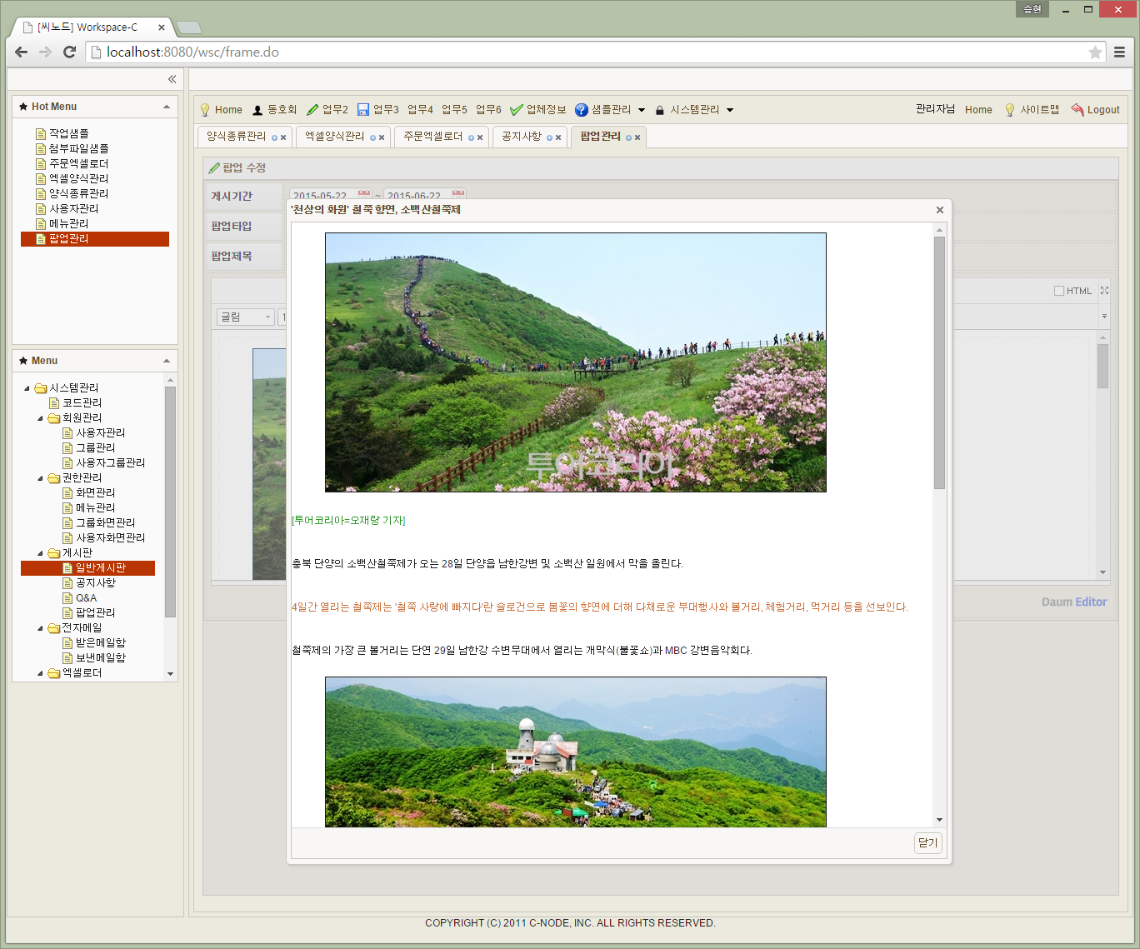
## 공지사항 (페이지 이동형 게시판)

* 레이어 팝업형태로 편집 창이나 상세조회 창이 뜨는 형태의 일반게시판과 달리 편집 또는 상세보기 시에 페이지가 이동하는 형태의 게시판에 대한 요청이 있어 페이지 이동형태의 게시판을 공지사항을 예제로 개발.
* 만약 레이어 팝업 형태의 게시판을 구현하고자 한다면 일반게시판을 참고하도록 하며, 페이지 이동형 게시판을 구현하고자 한다면 공지사항을 참고 하도록 한다.

## 팝업관리

* 메인페이지에 이벤트성 팝업을 표시하고자 할 때 사용되는 기능으로 페이지 이동형 게시판 형태의 구성을 하고 있다.
* 팝업타입은 코드관리에서 OPEN\_TYPE(팝업오픈타입) 의 항목을 나타내며, 순서대로 너비, 높이, 오픈시 X좌표, 오픈시 Y좌표의 값을 픽셀단위로 설정하여 사용하도록 한다.
* 미리보기 버튼을 클릭시엔 설정된 내용이 선택된 오픈타입에 따라 팝업창이 오픈된다.





## 개발 참고 샘플

* [샘플관리] – [예제모음] – [작업샘플] 에는 다음과 같은 기능의 예제가 구현되어 있으므로 개발시 참고하여 작업하도록 한다.

1. 멀티라인 코드
2. 콤보리셋 기능 및 콤보의 초기 로딩시 기본 텍스트 적용방법
3. 일반 텍스트 박스와 날짜 박스의 포커스 방법
4. 텍스트박스, 콤보박스 등의 Dynamic 행 추가 방법

* [샘플관리] – [예제모음] – [첨부파일샘플] 에는 다음과 같은 기능의 예제가 구현되어 있으므로 개발시 참고하여 작업하도록 한다.

1. 첨부파일 선택하여 업로드
2. 첨부파일 업로드 후에 화면에 목록으로 표시하는 방법

* [샘플관리] – [예제모음] – [주문엑셀로더] 에는 다음과 같은 기능의 예제가 구현되어 있으므로 개발시 참고하여 작업하도록 한다.

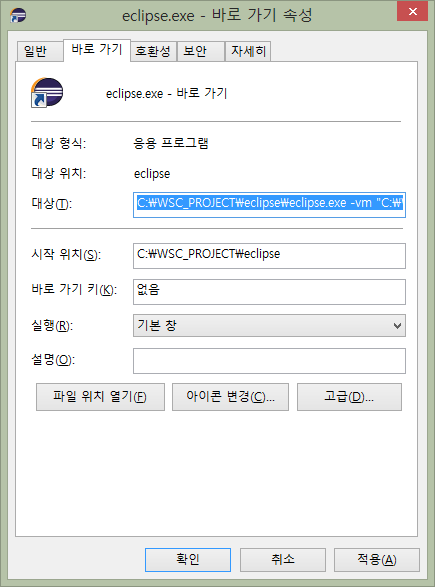
1. [엑셀양식관리]에서 설정한 맵핑정보대로 해당 양식의 엑셀파일을 업로드하여 목록으로 보여주고 저장버튼 클릭시 서버로 전달되는 예제.

# 프로젝트 개발환경 설정방법

## 개발환경 설치

* C:\ 경로에 WSC\_PROJECT.zip 파일을 복사한다.
* WSC\_PROJECT.zip 파일의 압축을 해당 경로명으로 압축을 풀면 C:\WSC\_PROJECT 경로가 생성된다.
* C:\WSC\_PROJECT\eclipse\eclipse.exe 파일 선택 후 오른쪽마우스 버튼을 눌러 바탕화면에 바로가기 아이콘을 생성한다.
* 바탕화면의 eclipse 아이콘을 선택 후 오른쪽 마우스를 눌러 속성을 선택한다.
* 속성의 바로 가기 탭에 있는 대상에 해당되는 항목을 그림과 같이 수정한다.

대상: C:\WSC\_PROJECT\eclipse\eclipse.exe -vm "C:\WSC\_PROJECT\java\jdk1.6.0\_30\_x64\jre\bin\javaw.exe"



* 바탕화면의 eclipse 아이콘을 더블 클릭하면 이클립스가 실행된다.

## 이클립스 내에 SVN 설정 및 소스 다운로드

* “SVN Repository Exploring” 선택 후 OK 를 클릭한다.
* 열린 “SVN Repositories” 탭 안에서 오른쪽 마우스 클릭 -> New -> Repository Location 선택한 후

URL: http://1.11.3.253:1024/svn/wsc

계정: 본인의 SVN 계정

을 입력한 후 Finish 를 클릭하면 해당 탭에 정의한 SVN이 생성된다.

(SVN 계정 정보는 관리자에게 문의하도록 한다.)

* http://1.11.3.253:1024/svn/wsc/tags/wsc-dev/wsc-1.0 프로젝트를 선택 한 후 오른쪽 마우스 클릭 -> Check Out 을 선택하면 해당 프로젝트 소스가 나의 로컬 경로로 복사된다.
* Java EE -> Project Explorer -> wsc-1.0 이 보이면 최종적으로 소스의 Import가 완성된다.

## 이클립스 내의 WAS 설정

* 하단에 보이는 Servers 탭 안에서 오른쪽 클릭 -> New -> Server를 선택한다.
* Apache -> Tomcat v7.0 Server 를 선택 후 Next를 클릭한다.
* wsc-1.0 선택 -> Add 버튼 클릭 -> Finish를 클릭한다.

## 프로젝트 환경 설정

* 파일 업로드용 Resource 경로 생성

C:\WSC\_PROJECT\workspace\wsc-1.0\upload

C:\WSC\_PROJECT\workspace\wsc-1.0\upload\real

C:\WSC\_PROJECT\workspace\wsc-1.0\upload\temp

C:\WSC\_PROJECT\workspace\wsc-1.0\upload\excel

* {프로젝트}/src/main/resources/app.properties 정보 변경

# FILE UPLOAD Physical Path

file.upload.temp=C:/WSC\_PROJECT/workspace/wsc-1.0/upload/temp

file.upload.real=C:/WSC\_PROJECT/workspace/wsc-1.0/upload/real

# Excel Download Template Path

excel.template.path=C:/WSC\_PROJECT/workspace/wsc-1.0/src/main/webapp/WEB-INF/templates

excel.template.temp=C:/WSC\_PROJECT/workspace/wsc-1.0/upload/excel

## WAS 실행 및 첫 페이지 구동 결과 확인

* 하단에 보이는 Servers 탭 - Tomcat v7.0 Server at localhost 선택 -> 오른쪽 클릭 -> Start 선택
* 브라우저에서 다음의 URL을 클릭 후 로그인 페이지가 나오면 WAS는 정상적으로 구동되었음이 확인된다.
* User ID / Password를 입력 후 로그인 버튼을 클릭 -> 메인 페이지가 나오는 것을 확인하면 완료된다.

(로그인 정보는 관리자에게 문의하도록 한다.)

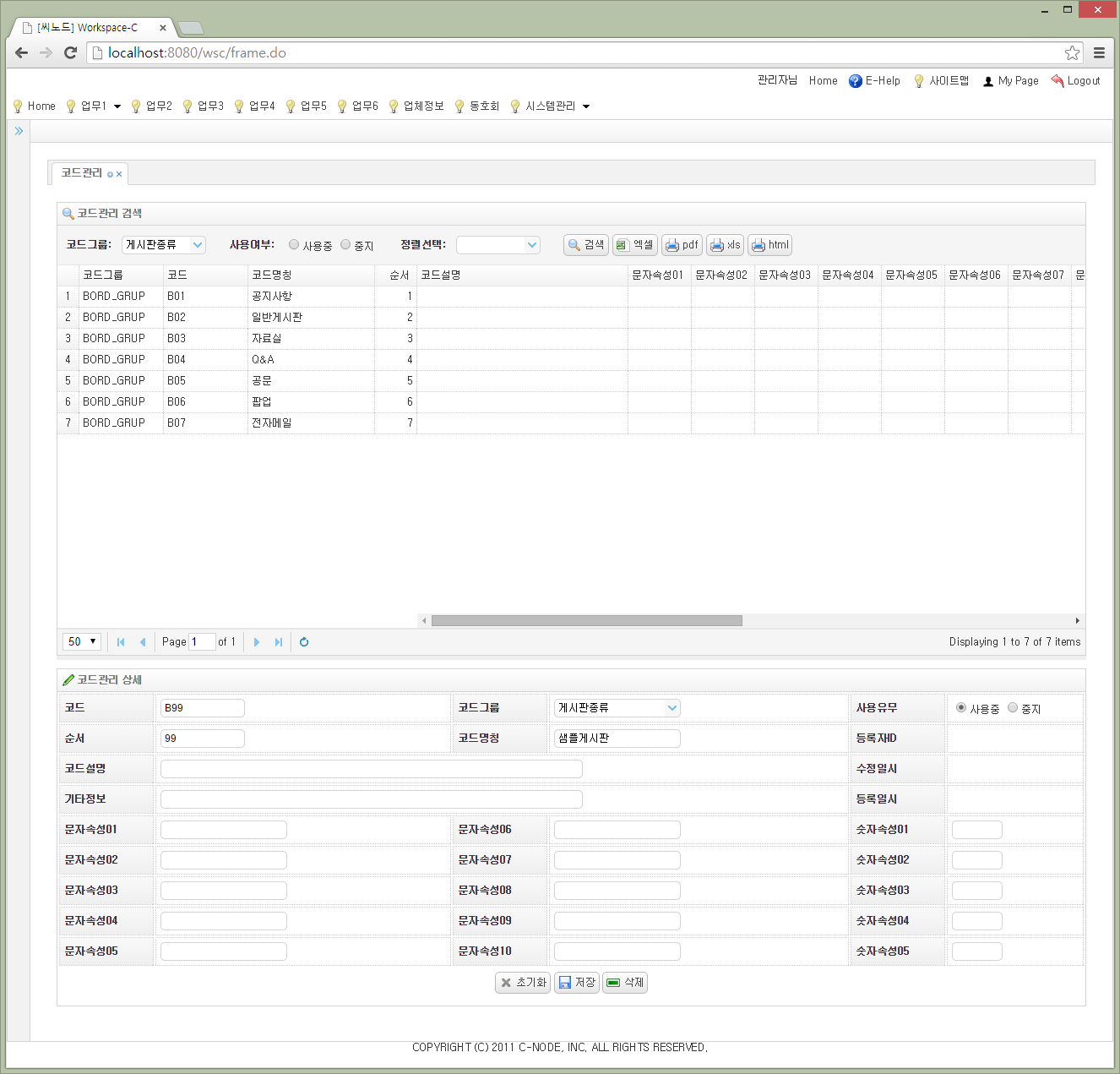
# 샘플게시판 개발 가이드

일반게시판의 유형에 해당하는 샘플게시판을 단계별로 따라가며 개발해보는 가이드이다. 다음의 규칙에 따라 샘플게시판을 개발해 보도록 한다.

|  |  |
| --- | --- |
| **항목** | **값** |
| 게시판 그룹코드 | B99 |
| 게시판 명칭 | 샘플게시판 |
| 첨부파일 경로 | /board99 |
| 패키지 명칭 | com.wsc.board |
| 클래스 기본명칭 | Sample |
| URL 기본경로 | /common/board/sample/ |

## 게시판 그룹 추가

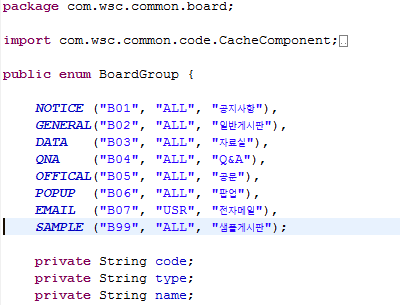
* 관리자로 시스템 로그인 후에 시스템관리 – 코드관리로 이동하여 코드그룹을 “게시판종류”를 조건으로 검색한다. 등록 폼에서 코드그룹을 “게시판종류”로 선택하고, 코드, 순서, 코드명칭, 사용유무를 입력한 후, 저장버튼을 클릭한다.



**입력항목**

* 자바소스폴더에서 com.wsc.common.board.BoardGroup클래스에 위에 입력한 코드에 해당되는 정보를 다음과 같이 입력한 후 저장한다.

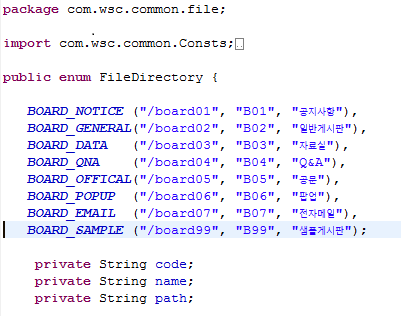
***SAMPLE*** ("B99", "ALL", "샘플게시판")



## 첨부파일 정보 추가

* 자바소스폴더에서 com.wsc.common.file.FileDirectory 클래스에 다음과 같이 입력한 후 저장한다.

***BOARD\_SAMPLE*** ("/board99", "B99", "샘플게시판")

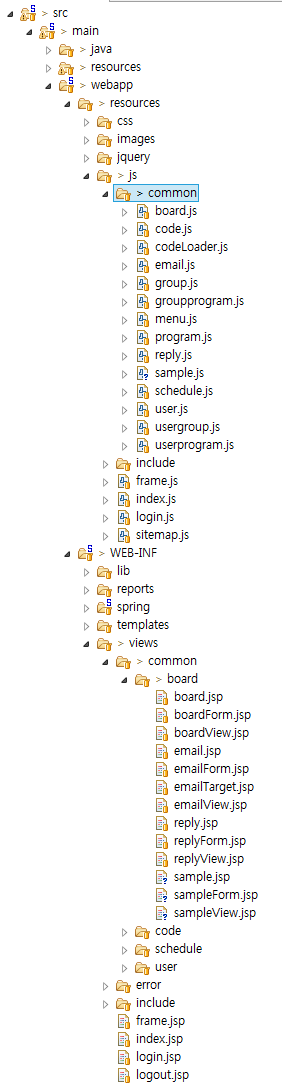


## 게시판 클래스 생성

* 이클립스에서 com.wsc.common.board.BoardController 클래스를 Copy하여 해당 패키지에 Paste한 후에 SampleController 로 명명한다.
* SampleController에서 선언부의 @RequestMapping “/common/board” 를 “/common/board/sample”로 수정한다.
* SampleController의 BoardGroup에 해당되는 부분을 BoardGroup.GENERAL에서 BoardGroup.SAMPLE로 변경한다.
* SampleController의 관리화면오픈 메서드의 @RequestMapping “/board.do”를 “/sample.do”로 수정한다.
* SampleController의 편집화면오픈 메서드의 “boardForm” 문자열을 “sampleForm” 문자열로 수정한다.
* SampleController의 상세화면오픈 메서드의 “boardView” 문자열을 “sampleView” 문자열로 수정한다.



## JSP and JavaScript 생성

* 이클립스에서 src/main/webapp/WEB-INF/views/common/board/ 경로안의 board.jsp, boardForm.jsp, boardView.jsp를 Copy 하여 동일경로에 Paste 한 후 각각 sample.jsp, sampleForm.jsp, sampleView.jsp 로 명명한다.
* 이클립스에서 src/main/webapp/resources/js/common/ 경로 안의 board.js 를 Copy 하여 동일경로에 Paste 한 후 각각 sample.js로 명명한다.
* sample.jsp를 편집기에서 열고 업무스크립트를 *"/resources/js/common/board.js" => "/resources/js/common/sample.js"*로 수정한다.
* sample.js를 편집기에서 열고 consts.url 항목의 URL을 Controller에서 정의한 URL로 모두 변경한다.

|  |  |
| --- | --- |
| URL | 변경 전 |
| search:  excel:  select:  remove:  save:  form:  view: | getUrl("/common/board/search.json")  getUrl("/common/board/download.do")  getUrl("/common/board/select.json")  getUrl("/common/board/delete.json")  getUrl("/common/board/save.json")  getUrl("/common/board/form.do")  getUrl("/common/board/view.do") |

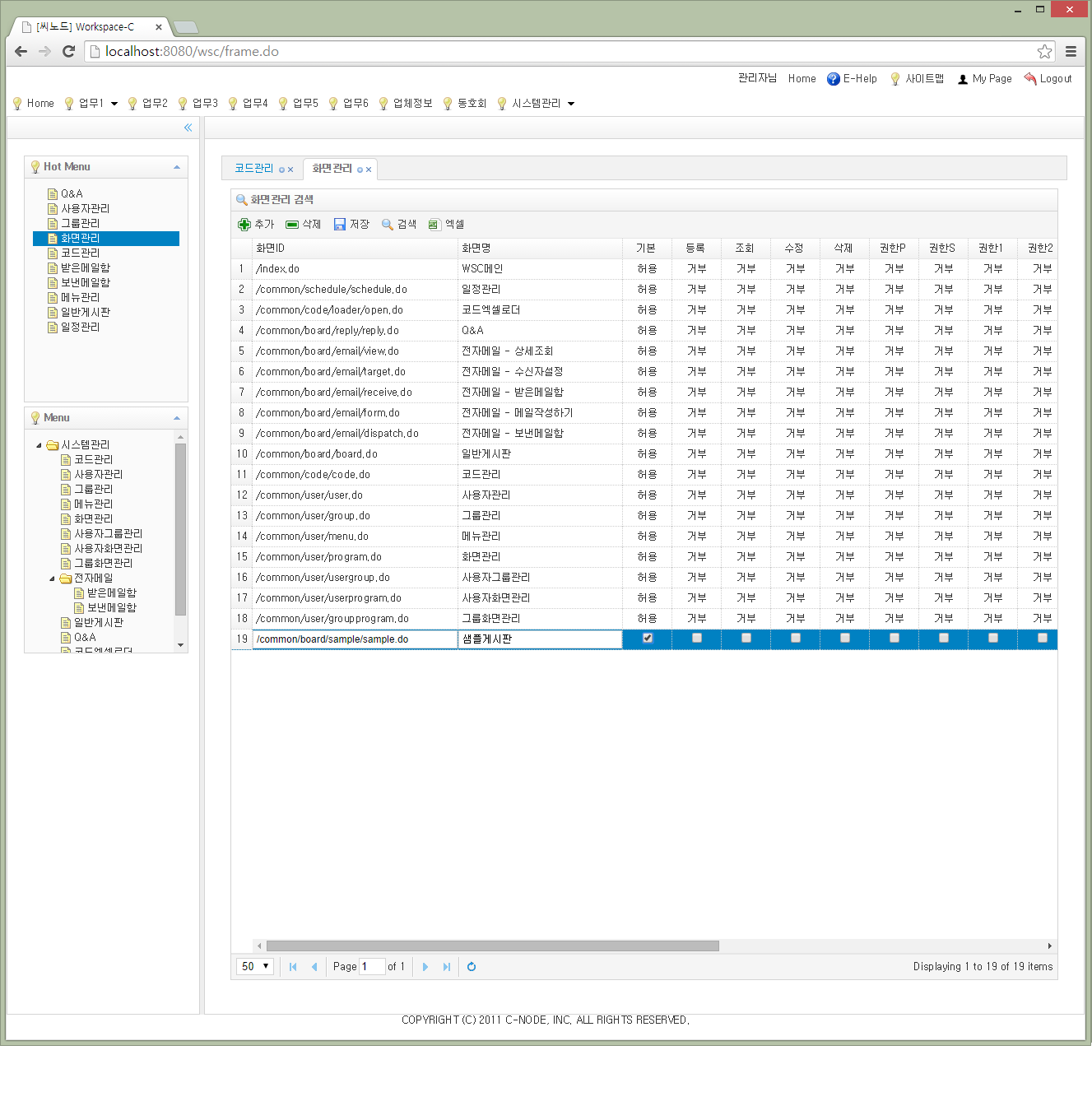
|  |  |
| --- | --- |
| URL | 변경 후 |
| search:  excel:  select:  remove:  save:  form:  view: | getUrl("/common/board/sample/search.json")  getUrl("/common/board/sample/download.do")  getUrl("/common/board/sample/select.json")  getUrl("/common/board/sample/delete.json")  getUrl("/common/board/sample/save.json")  getUrl("/common/board/sample/form.do")  getUrl("/common/board/sample/view.do") |

## WAS 실행

* 프로젝트 개발환경 설정방법을 참고한다.

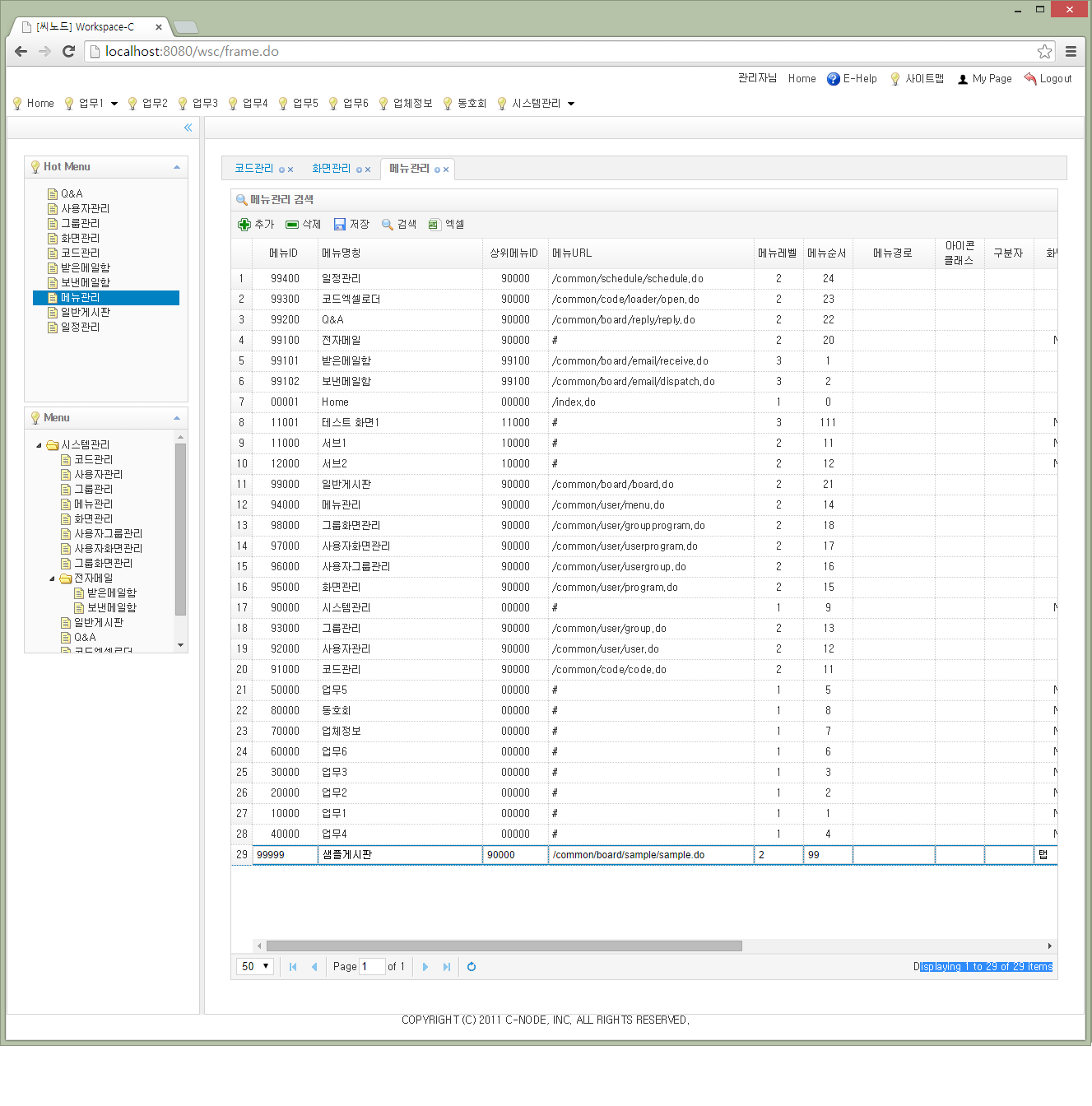
## 메뉴 생성 및 권한 설정

* 웹 화면에서 시스템관리 – 화면관리로 이동한 후 추가 버튼을 눌러 게시판의 화면정보를 참고하여 관리화면 URL을 화면으로 등록한다.

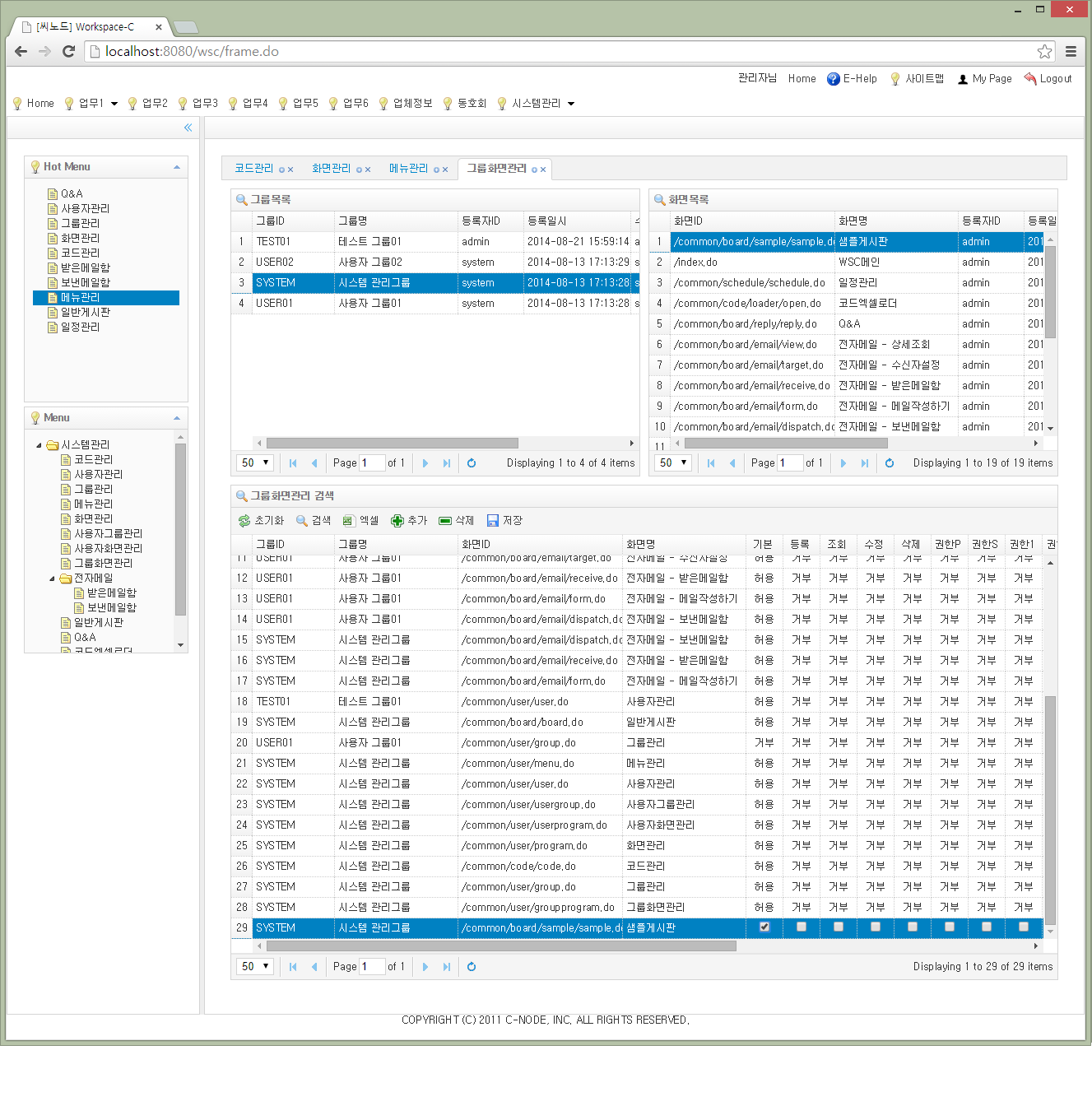


* 웹 화면에서 시스템관리 – 메뉴관리로 이동한 후 추가 버튼을 눌러 게시판의 메뉴정보를 참고하여 관리화면 URL을 메뉴로 등록한다.

(필수입력: 메뉴ID, 메뉴명칭, 상위메뉴ID, 메뉴URL, 메뉴레벨, 메뉴순서, 화면타입, 활성화 여부)



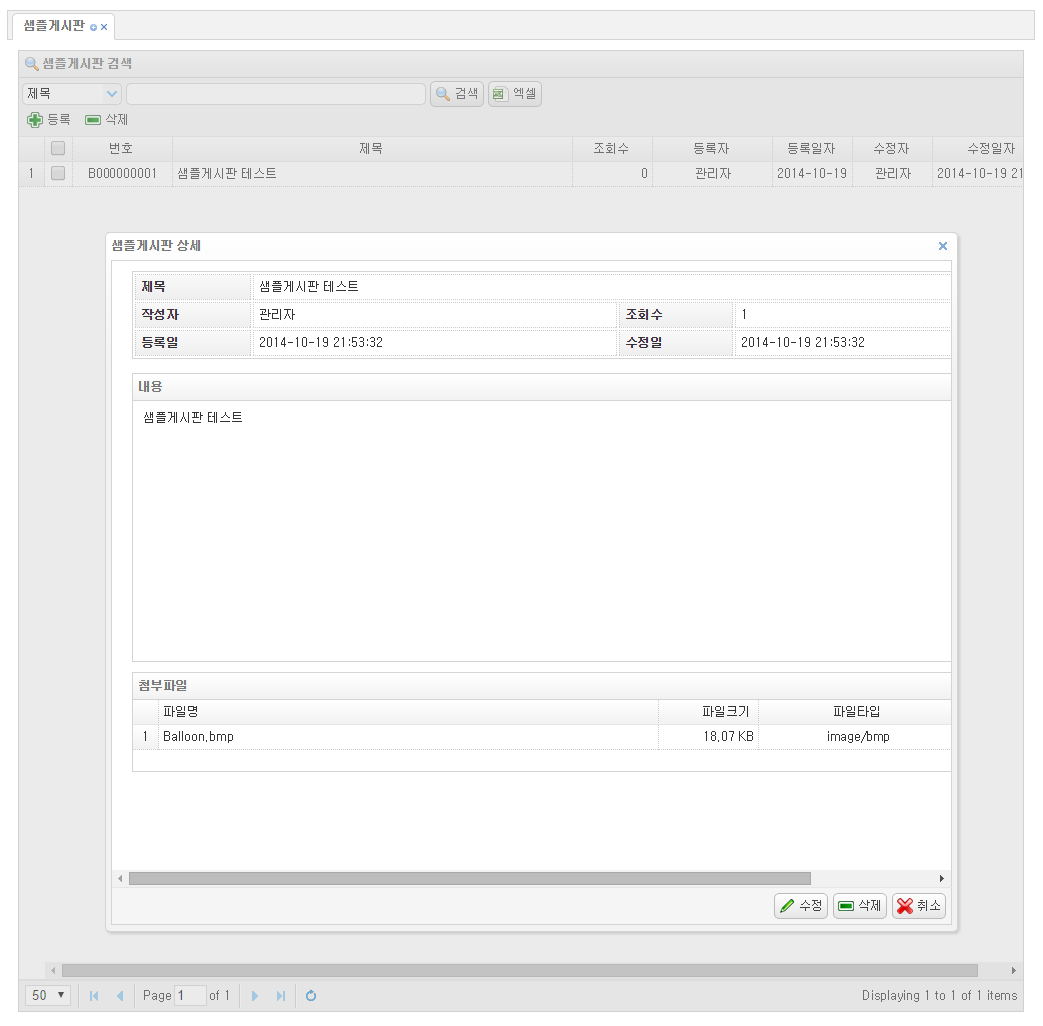
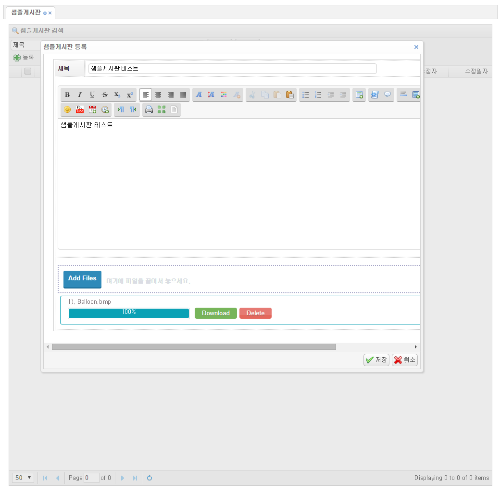
* 웹 화면에서 시스템관리 – 그룹화면관리로 이동한 후 [그룹목록]에서 “시스템관리그룹”을 선택하고, [화면목록]에선 “샘플게시판”을 선택한 후 [그룹화면관리] 목록의 추가 버튼을 누른 다음, 추가된 항목의 기본항목을 허용으로 변경한 후 저장버튼을 클릭하여 시스템관리그룹에게 생성한 게시판의 접근권한을 부여한다.



* 처리완료 후 로그아웃 후에 다시 로그인해야 추가된 메뉴를 확인할 수 있다.

## 결과 확인

* 웹 화면에서 시스템관리 – 샘플게시판으로 이동하여 게시 글을 등록하고, 등록한 글을 검색해본다.



* 정상적으로 실행되면 해당 샘플게시판의 생성이 완료된다.

# 이후 개발 과제

## 현재까지 구현된 기능

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **분류1** | **분류2** | **분류3** | **작업내용** |
| 1 | 프레임워크 | Configuration | Spring | 프레임워크 개발환경 설정 |
| 2 | 프레임워크 | Configuration | Spring | 프레이워크 통합 구성 |
| 3 | 프레임워크 | Configuration | Mybatis | 프레임워크 Mybatis 설정 및 구성 |
| 4 | 프레임워크 | Configuration | Logging | 프레임워크 Log4j 설정 및 구성 |
| 5 | 프레임워크 | Configuration | Mybatis | 프레임워크 트랜젝션 설정 및 어노테이션 처리 |
| 6 | 프레임워크 | Configuration | Spring | 프레임워크 공통 상위클래스 정의 |
| 7 | 프레임워크 | Configuration | Exception | Exception Handling 및 Logging 처리 |
| 8 | 프레임워크 | Configuration | Resource | 메세지리소스 적용(로케일 적용) |
| 9 | 프레임워크 | Configuration | Locale | Locale 변경 처리 |
| 10 | 프레임워크 | Configuration | Cache | ehcache 설정 및 구성 |
| 11 | 프레임워크 | Configuration | Cache | 코드데이터 캐쉬 구성 |
| 12 | 프레임워크 | Configuration | Security | 로그인, 로그아웃, 사용자로깅 |
| 13 | 프레임워크 | Configuration | Security | 인증 및 권한처리 구성 |
| 14 | 프레임워크 | Configuration | Security | SYS\_ID 기본설정 구성(세션처리) |
| 15 | 프레임워크 | Configuration | Security | 비밀번호 변경횟수 누적 및 리셋 |
| 16 | 프레임워크 | Common | EasyUI | EasyUI 설정 및 적용 |
| 17 | 프레임워크 | Common | Common | 공통 INCLUDE 생성 |
| 18 | 프레임워크 | Common | EasyUI | 상단메뉴, 좌측메뉴 개발 |
| 19 | 프레임워크 | Common | EasyUI | 기본 템플릿 및 화면 레이아웃 설정 |
| 20 | 프레임워크 | Common | EasyUI | 그리드 기본 설정 |
| 21 | 프레임워크 | Common | EasyUI | 그리드 편집 설정 |
| 22 | 프레임워크 | Common | EasyUI | 폼패널 및 팝업폼 설정 및 편집기능 구현 |
| 23 | 프레임워크 | Common | EasyUI | 코드 연동된 콤보 검색 설정 |
| 24 | 프레임워크 | Configuration | Mybatis | 프레임워크 서비스 트랜젝션 테스트 |
| 25 | 프레임워크 | Configuration | Security | 접근권한VIEW 적용 및 권한적용 테스트 |
| 26 | 프레임워크 | Common | Pagination | 페이징 처리 설정 및 그리드 페이징 연동 |
| 27 | 공통개발 | 시스템관리 | 사용자관리 | 사용자관리 개발작업 |
| 28 | 공통개발 | 시스템관리 | 그룹관리 | 그룹관리 개발작업 |
| 29 | 공통개발 | 시스템관리 | 코드관리 | 코드관리 개발작업 |
| 30 | 공통개발 | 시스템관리 | 메뉴관리 | 메뉴관리 개발작업 |
| 31 | 공통개발 | 시스템관리 | 화면관리 | 화면관리 개발작업 |
| 32 | 공통개발 | 시스템관리 | 사용자그룹관리 | 사용자그룹관리 개발작업 |
| 33 | 공통개발 | 시스템관리 | 사용자화면관리 | 사용자화면관리 개발작업 |
| 34 | 공통개발 | 시스템관리 | 그룹화면관리 | 그룹화면관리 개발작업 |
| 35 | 프레임워크 | Common | Web Editor | 웹에디터 플러그인 추가적용 |
| 36 | 프레임워크 | Common | File Control | 파일업로드 플러그인 추가적용 |
| 37 | 프레임워크 | Common | File Control | 첨부파일 업로드 개발작업 |
| 38 | 프레임워크 | Common | File Control | 첨부파일 다운로드 개발작업 |
| 39 | 프레임워크 | Common | Pagination | 페이징처리 테스트 및 수정 |
| 40 | 프레임워크 | Common | EasyUI | 콤보그리드 적용(사용자관리 예제) |
| 41 | 공통개발 | 시스템관리 | 게시판 | 일반게시판(게) |
| 42 | 공통개발 | 시스템관리 | 사용자관리 | 관리자의 사용자별 자동로그인 기능 |
| 43 | 프레임워크 | Common | Menu | 메뉴 검색시 권한연동 설정 |
| 44 | 프레임워크 | Common | Excel Control | 엑셀다운로드 개발 |
| 45 | 프레임워크 | Common | Excel Control | 엑셀다운로드 적용 |
| 46 | 공통개발 | 시스템관리 | 게시판 | 전자메일-개인주소록 관리 및 게시대상 설정 |
| 47 | 공통개발 | 시스템관리 | 게시판 | 전자메일-그룹주소록 관리 및 게시대상 설정 |
| 48 | 공통개발 | 시스템관리 | 게시판 | 전자메일-수신,발신,보관,발송 |
| 49 | 프레임워크 | Common | Common | 그리드의 헤더 정렬 적용 |
| 50 | 공통개발 | 시스템관리 | 게시판 | Q&A(게) - 답변게시판 트리그리드 적용 |
| 51 | 공통개발 | 초기화면 | 사이트맵 | 사이트맵 구현 - 탭없는페이지 샘플 |
| 52 | 공통개발 | 초기화면 | 핫메뉴 | 즐겨찾기메뉴(HOTMENU) - 컨텍스트메뉴 설정 |
| 53 | 프레임워크 | Common | Excel Control | 엑셀로더 개발 |
| 54 | 공통개발 | 시스템관리 | 코드엑셀로더 | 코드엑셀로더 개발작업 |
| 55 | 프레임워크 | Common | EasyUI | 그리드 다중정렬,로컬정렬,콤보정렬 샘플작업 |
| 56 | 공통개발 | 시스템관리 | 일정관리 | 일정관리(개인,공유일정) |
| 57 | 공통개발 | 시스템관리 | 일정관리 | 일정관리(개인,공유일정) - 공유대상설정 |
| 58 | 프레임워크 | Common | EasyUI | 단계콤보 작업 |
| 59 | 프레임워크 | Common | Printing | iReport를 활용한 출력기능 구현 |
| 60 | 공통개발 | 시스템관리 | 공지사항 | 공지사항(게) |
| 61 | 공통개발 | 시스템관리 | 공지사항 | 공지사항(게) - 게시기간 설정 |
| 62 | 공통개발 | 시스템관리 | 팝업관리 | POPUP(게) |
| 63 | 공통개발 | 초기화면 | 팝업 | POPUP(게) - POPUP 메인공지 |
| 64 | 공통개발 | 시스템관리 | 팝업관리 | POPUP(게) – 미리보기 |
| 65 | 공통개발 | 시스템관리 | 엑셀로더 | 자동맵핑 엑셀로더 개발 |
| 66 | 프레임워크 | Common | 샘플예제 | 개발참고용 샘플예제 개발 |
| 67 | 프레임워크 | Common | 웹에디터 | 웹에디터 확장(다음에디터, CK에디터) |
| 68 | 공통개발 | 시스템관리 | 게시판 | 페이지 이동형 게시판 개발 |
| 69 | 프레임워크 | Common | EasyUI | 화면 기본 레이아웃 변경 |
| 70 | 프레임워크 | Common | Common | 메뉴의 아이콘 설정 기능 개발 |
| 71 | 프레임워크 | Common | Common | 디자인 요청용 샘플HTML 및 가이드 문서 |
| 72 | 프레임워크 | Common | Common | 탭패널형과 일반형태의 자동전환 기능 |

## 아직 개발되지 않은 기능

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **분류1** | **분류2** | **분류3** | **작업내용** |
| 1 | 프레임워크 | Common | Security | 버튼별 권한 설정 |
| 2 | 공통개발 | 시스템관리 | 게시판 | 공문 수/발신 이력조회 |
| 3 | 공통개발 | 시스템관리 | 게시판 | 공문관리(게) |
| 4 | 공통개발 | 시스템관리 | 게시판 | 긴급통보서(게) |
| 5 | 공통개발 | 시스템관리 | 게시판 | 도움말(게) |
| 6 | 공통개발 | 시스템관리 | 게시판 | 도움말(게) - 개인도움말(E-Help) |
| 7 | 공통개발 | 시스템관리 | 게시판 | 자료실(게) |
| 8 | 공통개발 | 시스템관리 | 게시판 | 통합게시판 관리 |
| 9 | 공통개발 | 시스템관리 | 용어관리 | 용어관리 |
| 10 | 공통개발 | 시스템관리 | 용어관리 | 화면/용어관리 |
| 11 | 공통개발 | 시스템관리 | 코드관리 | 코드이력관리 |
| 12 | 공통개발 | 시스템관리 | 코드관리(BP) | 코드관리(BP) |
| 13 | 공통개발 | 시스템관리 | 로그인현황 | 로그인현황(사용자로그 조회) |
| 14 | 공통개발 | 시스템관리 | 배치현황 | 배치작업현황 |
| 15 | 공통개발 | 초기화면 | MyPage | MyPage 기능 구현 |
| 16 | 공통개발 | 초기화면 | 게시판 | 공문/게시판 |
| 17 | 공통개발 | 초기화면 | 알람 | 대쉬보드(Alarm) |
| 18 | 공통개발 | 사용자공통 | 동호회관리 | 동호회관리(회원관리,게시판,앨범) |
| 19 | 공통개발 | 사용자공통 | 업체관리 | 업체관리 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## 보완 필요 사항

* 화면별 권한 처리에 대해 좀더 정밀한 테스트와 보완 필요.
* URL 유형별 에러메세지 표시에 대한 세부적인 처리 필요.
* 일정관리에서 전체적으로 적용될 Work Calendar에 대한 데이터 셋팅 및 그에 따른 보완 필요.
* 인증시 비밀번호 실패횟수에 따른 인증 제한 기능 필요.
* iReport 부분에 대한 세부적인 설정 및 출력 연동 필요.
* 세부적인 화면 웹디자인 변경 및 적용 필요.

# 프레임워크 개선 사항

## 1차 개선 사항 (2015년04월 – 2015년06월)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **번호** | **요청일** | **보완요청사항** | **처리내역** | **처리구분** |
| **1** | **2015-04-13** | IE8에서 datebox 날짜 선택 불가하며 임의 입력시 현재날짜로 저장됨. | datebox의 parser 함수(business.js)의 추가에 따른 버그였던 것으로 확인됨.  business.js 수정완료 | 수정 |
| **2** | **2015-04-13** | 3-1) combobox 데이터 맨앞에 특정 문구(선택,전체,..) 처리 | combobox의 옵션 중 prompt 라는 옵션은 콤보박스의 기본문구를 설정하는 기능을 하며, 별도의 선택행위가 필요한 경우를 위해 combobox의 loader 함수에 "first:{value:'', text:'전체'}" 와 같이 첫번째 항목에 옵션을 추가할 수 있는 first 라는 옵션을 설정하도록 해당 기능을 추가함.  common.js loader 함수내 기능추가완료. [샘플관리]-[샘플메뉴]-[작업샘플] 화면에 샘플예제를 참조할것. | 기능추가 |
| **3** | **2015-04-13** | 3-2) textbox, datebox 의 $("#id").focus(); 불가 | input box 를 포커싱하는 방법은 일반적인 jquery 사용법으로는 불가능함. 예를 들어 텍스트박스의 경우엔 $("#id").textbox("textbox").focus(); 와 같이 EasyUI의 함수를 이용하여야 함.  [샘플관리]-[샘플메뉴]-[작업샘플] 화면에 샘플예제를 정의함. | 샘플추가 |
| **4** | **2015-04-13** | 3-3) 동적으로 textbox 및 combobox 생성 불가 | 동적인 생성방법은 기본적으로 jquery에서 사용하는 방법과 동일하나, 다만 textbox나 combobox 의 객체를 스크립트상에서 생성해줘야 함.[샘플관리]-[샘플메뉴]-[작업샘플] 화면에 샘플예제를 참조할것. | 샘플추가 |
| **5** | **2015-04-13** | 3-4) IE9에서 파일업로드 기능 작동 불가 | 확인해본 결과 Drag & Drop 은 IE9에선 사용이 불가능함. 해당 플러그인의 데모화면도 마찬가지로 안되는 것으로 보아 플러그인의 버그로 추정됨. 또한 첨부파일이 업로드되지 않는 것으로 확인되었으며, 이는 IE9의 버그에 따른 것으로 업로드한 결과값을 json으로 내려줄때 이를 다운로드로 처리하는 문제로 인해 첨부파일이 등록되지 않음. 이부분은 우회하는 방법을 찾아 해결함.  우회하는 방법은 Content Type을 application/json 이 아닌 text/html 로 결과가 내려가도록 수정하였음. FileController.java 수정완료 | 수정 |
| **6** | **2015-04-13** | 3-5) textbox 멀티라인 입력시 DB에 입력된 대로 저장안됨. | 기본적으로 멀티라인 입력은 textarea와 동일한 방식으로 동작됨. 다만 DB 저장시 줄바꿈 등의 특수문자를 태그로 변경하거나 하는 작업이 필요한 경우엔 별도로 변환하는 코딩이 필요하며 이는 textarea 또한 마찬가지임.  [시스템관리]-[일정관리] 화면에서 일정등록시 내용 등의 항목은 textbox 의 multiline 옵션을 사용하여 정의되어 있으므로 이를 참조하기 바람. | 부연설명 |
| **7** | **2015-04-13** | 4-1) 상단메뉴의 전구아이콘 교체 불가 | 개발가이드 문서의 보완필요사항에 언급되었던 부분으로 해당 시점엔 구현되지 않은 기능이었음. 현재 메뉴등록시 아이콘클래스 항목이 입력되어 있으면 해당 클래스가 적용되도록 widget.js 에 기능을 추가 작업함.  widget.js 기능추가완료. | 기능추가 |
| **8** | **2015-04-13** | 6-1) 최상단 우측의 로그아웃버튼이 메인페이지에서 보이지 않는 문제 | 메인페이지 로딩시 사용자정보가 ModelMap 에 담기지 않아 생기는 현상임. CommonController 의 index 메서드에 해당 정보가 담기도록 수정함. 사이트맵 화면도 동일한 현상이 있어 같이 수정됨.  CommonController.java 수정완료 | 수정 |
| **9** | **2015-04-13** | 5-1) 게시판의 웹에디터를 "다음에디터"로 변경 처리 | 기존은 SCEDITOR 로 적용되어 있음.  **다음의 3가지의 에디터를 선택적용 가능하도록 추가작업함. 1) SC : SC에디터 2) CK : CK에디터 3) DM: 다음에디터 해당 선택은 jeditor.js 에서 정의되며, 에디터를 쓰는 프로그램에 일괄 적용됨. 단, 모든 에디터가 IE8에선 실행불가함.** | 기능추가 |
| **10** | **2015-04-23** | 8) 로그인 안되었는데 main화면에 URL로 바로 접근가능(http://127.0.0.1:8080/) | 로그인 체크 및 권한을 관리하기 위한 URL이 화면관리에 등록되어야 하는데 "/"에 해당되는 URL이 미등록되어서 생기는 문제이나 해당 URL은 "/index.do"와 동일하므로 동일한 URL로 맵핑되어 체크되도록 수정함.  SecurityInterceptor.java 수정완료 | 수정 |
| **11** | **2015-04-23** | 9) 메뉴 진입시 스크립터 에러(SCRIPT5007,5009)  - IE 11에서 디버그 모드 확인 결과 | EasyUI의 탭기능에 버그가 있어 생기는 문제로서 해당 사이트에서 패치된 소스를 받아 INCLUDE 함.(jquery-easyui-1.4-patch.zip)  jquery-easyui-1.4-patch.zip 를 다운받아 압축해제후 INCLUDE 함. | 패치적용 |
| **12** | **2015-04-23** | 3) 화면내 이동시 화면이 window.location.href 가 사용가능여부 | self.location.href 를 사용하면 해당 페이지내에서 이동이 가능함.  공지사항 개발예제를 참고할것. | 부연설명 |
| **13** | **2015-04-23** | 페이지 이동형 게시판 예제 개발 요청 | 페이지 이동형 게시판 예제 개발 요청  공지사항 개발예제 생성페이지 이동형 게시판으로 개발됨. | 기능추가 |
| **14** | **2015-04-23** | 2) 파일업로드 및 다운로드기능 공통 메소드 추가 | 가장 기본적인 파일 업로드 기능 개발 요청함.  [샘플관리] - [샘플메뉴] - [첨부파일샘플] 화면의 소스 및 예제 참조할것 | 기능추가 |
| **15** | **2015-04-23** | 4) 메인 컨덴츠 영역 방식 분기화 - 텝, 일반 화면전환 | 설정에 따라 탭과 일반 화면 사이를 전환할수 있는 기능 요청: 현재 가능한지 여부에 대한 판단이 필요하며, 그에 따라 추후 일괄 수정 작업이 필요할 것으로 보여짐.  **상단,하단 INCLUDE 페이지 적용** 및 메뉴의 링크방식을 변경함.해당 설정은 **app.properties 의 usable.tabPanel = true** 에서 설정되며, 해당 설정에 따라 모든페이지에 적용된다. | 기능추가 |
| **16** | **2015-04-23** | 엑셀 업로드(설명,예제 : G마켓 엑셀 양식)  : 칼럼 맵핑 기능을 포함한 엑셀양식에 따른 엑셀파일 업로드 기능 개발 요청. | 1) 양식종류관리 개발  : 주문 또는 출고 등의 양식의 그룹과 맵핑할 변수명을 정의하는 부분으로 해당 맵핑변수를 대상테이블의 칼럼에 대입하여 저장처리 기능을 구현하도록 한다. 2) 엑셀양식관리 개발  : 각 양식그룹에 해당하는 종류(오픈마켓)별 엑셀양식을 등록하고 개별 칼럼을 설정한다. 이 부분에서 1)번에서 정의한 맵핑변수와 개별 칼럼을 맵핑시키도록 한다. 3) 주문엑셀로더 개발  : 엑셀양식의 샘플로서 양식에 맞는 엑셀파일을 업로드하고 해당 데이터를 읽어 화면에 그리드형태로 display 하는 기능까지 개발되어 있으며, 저장버튼 클릭 후에 실제 저장을 처리하는 로직은 업무적인 요청에 따라 별도의 개발이 필요하다. | 기능추가 |
| **17** | **2015-04-23** | 7) 팝업관리 추가(설명필요,팝업좌표,사이즈,게시기간) | 게시판 형태의 팝업관리 기능 개발 및 메인의 팝업로딩 기능 추가 | 기능추가 |
| **18** | **2015-05-22** | 그리드에서 idField에 해당하는 항목의 편집시 datebox 임에도 validatebox로 표시되는 문제 | business.js 에서 afterEdit 함수에서 무조건 validatebox 로 editor가 변경되도록 설정되어 있었음.  설정된 에디터가 적용되도록 business.js 의 afterEdit 함수를 수정함. | 버그수정 |
| **19** | **2015-05-22** | 그리드에서 특정칼럼 편집시 다른칼럼의 값이 셋팅되도록 하는 예제 요청 (예를 들어 코드를 입력하면 코드에 해당되는 명칭이 조회되어 옆의 칼럼에 자동입력되도록 하는 기능) | dps 프로젝트의 [품절등록] 화면 에서 상품코드 선택시 AJAX를 이용하여 상품명을 가져오는 로직의 샘플을 제공함. : 해당 처리를 위해서 상품코드,상품명은 모두 에디터가 있는 형태여야 하며, 그리드의 onBeginEdit 이벤트에 해당 처리로직을 바인딩함. | 샘플추가 |
| **20** | **2015-05-22** | 팝업관리 - 화면 오픈시 개발자도구에서 스크립트 오류가 발생하며 에디터 영역이 표시가 잘 되지 않습니다. | IE에서 간헐적으로 발생하는 스크립트 오류로서 SCEDITOR 일 경우에만 해당 문제가 발생되고 있음을 확인함.  SCEDITOR 에 국한한 문제이며 IE에서만 간헐적으로 발생하므로 에디터를 다음에디터로 변경함으로써 해당 문제를 우회하여 해결함. | 버그수정 |
| **21** | **2015-05-22** | 팝업관리 - 미리보기 기능 추가개발 요청 | 편집화면과 상세화면에서 미리보기 버튼을 두어 해당 버튼 클릭시 미리보기용 팝업이 띄워지도록 개발 작업함 | 기능추가 |
| **22** | **2015-05-22** | 화면 영역 확인 요청 : 여백이 많이 존재 하는데 전체적으로 꽉찬 느낌으로 변경이 가능 할까요?  EasyUI 의 ThemeBuilder 와 같은 형태로 Layout 변경을 요청함. | 해당 형태에 맞춰 CSS 와 INCLUDE 용 JSP 를 수정함. (common.css, head.jsp, foot.jsp, north.jsp, body.head.jsp, body.foot.jsp) | 수정개발 |
| **23** | **2015-06-02** | 레이아웃 변경  테마빌더 내의 중앙부분의 레이아웃과 같은 방식으로 재변경요망 | 이전버젼(메뉴가 최상단에 위치)으로 복귀 중앙패널의 border 및 titlebar 제거  : body.head.jsp, head.jsp, foot.jsp, body.foot.jsp, north.jsp 수정 | 수정 |
| **24** | **2015-06-02** | 메뉴바 스크롤 확인 | 최상단 메뉴의 자동스크롤 처리 common.css (overflow:hidden -> overflow: auto) | 수정 |
| **25** | **2015-06-02** | 공통코드의 useFlag(사용여부) 조건 설정  : 콤보에서 코드검색시 코드그룹외의 기타조건이 추가될 수 있는지 문의 | useFlag(사용여부) 조건이 조건비교상에 생략되어 있어 해당 조건을 추가함. 해당 조건을 사용하는 예제는 **"엑셀양식관리(loaderForm.js)"** 를 참고할것 | 버그수정 |
| **26** | **2015-06-02** | 그리드의 중복확인과 같은 LOOP Validation 예제 | "양식종류관리" 참고할것. : **loaderCode.js 의 doValidate** 함수 확인할것 : **common.js 의 jutils.duplicateObject** 함수 참고할것 | 확인 |
| **27** | **2015-06-02** | 코드관리에서 검색시 타 기능과 다르게 유달리 로딩이 느린 문제가 발생함. | 그리드의 칼럼항목 설정시 **width 지정**이 안되어 있으면 칼럼수가 많을 경우 로딩이 느려지는 문제가 발생함. 이에 따라 칼럼의 width 를 개별 지정해줌. code.jsp 참고할것. | 버그수정 |
| **28** | **2015-06-02** | 그리드 행추가시 idField 에디터 비활성 문제 | 기본설정된 에디터가 수정시에 제거됨으로서 추가시에 재생성되지 못하는 문제로서 별도로 해당 에디터를 KEEP 할수 있도록 수정. business.js 내의 beforeEdit 함수 수정 | 버그수정 |
| **29** | **2015-06-09** | 각 패널간, 프레임간 여백 간격 모두 제거할 것 | 패널간 여백 제거 해당 여백이 설정된 CSS 는 별도로 common.css 내에 주석처리함. | 수정 |
| **30** | **2015-06-09** | 좌측 영역 메뉴접기 아이콘으로 설정할것 | [메뉴닫기] 아이콘을 west.jsp 에 정의하여 좌측 영역에 타이틀바에 있는 메뉴접는 기능을 아이콘으로 구현. 좌측 영역의 타이틀바는 숨김 처리  (아이콘은 디자인 후에 대체하도록 할것) | 기능추가 |