|  |
| --- |
| 휴먼플러스 |
| Spring Java Web 개발 표준 지침서 |
| [초안] |

|  |
| --- |
| 2014-12-18 |

**< 목 차 >**

**1. 개발 표준 개요 ……………………………………………………………………………………………………………………………………….... 02**

**1.1 개발 표준 수립의 목적 …………………………………………………………………………………………………………………………. 02**

**2. 개발환경 …………………………………………………………………………………………………………………………………………………. 03**

**2.1 JDK 1.7 version 설치 및 환경설정 ............................................................................................................................. 03**

**2.2 Spring STS 설치 ................................................................................................................................................. 06**

**2.3 환경설정 …………………………………………………………………………………………………………………………………………….. 08**

**3. 분류코드 ……………………………………………………………………………………………………………………………………………….... 17**

**3.1 시스템 코드 : B2B ……………………………………………………………………………………………………………………………….. 17**

**4. 디렉토리 및 파일 …………………………………………………………………………………………………………………………………….. 17**

**4.1 프로그램 및 업무 속성에 따른 디렉토리의 분리 ……………………………………………………………………………………... 17**

**5. 자바 코딩 표준 ……………………………………………………………………………………………………………………………………….. 18**

**5.1 자바 소스파일 ………………………………………………………………………………………………………………………………….. 18**

**6. SQL …………………………………………………………….....................……………………………………………………………….. 24**

**6.1 SQL Map ………………………………………………………………………………………………………………………………….. 24**

**6.2 MyBatis ................................................................................................................................................... 25**

**7. 화면디자인 ………………………………………………………………………………...............…………………………………………….. 26**

**7.1 화면 소스파일 ………………………………………………………………………………………………………………………………….. 26**

**7.1 View Pattern ...................................................................................................................................... 28**

**7.2 View Pattern Common ...................................................................................................................................... 30**

**7.3 View Pattern 1 Grid Board ...................................................................................................................................... 37**

**7.4 View Pattern 2 Grid ...................................................................................................................................... 41**

**7.5 View Pattern Popup ...................................................................................................................................... 46**

**7.6 View Pattern 1Grid 1Form ........................................................................................................................................... 48**

**8. MVC Pattern Practice …………………………………………………………………………………………………………………………….. 50**

**1. 개발 표준 개요**

**1.1 개발 표준 수립의 목적**

**본 문서는 Spring Framework 기반 B2B 시스템 구축 프로젝트에서 프로그램 설계자와 개발자들에게 구현, 배포, 관리 등의 절차에 대한 지침을 제공하는데 목적이 있다.  
또한 코드 규약은 아래와 같은 이유로 개발자들이 반드시 지켜야 할 사항이다.**

**① 소프트웨어 생명의 80%가 유지보수에 소요된다**

**② 본래의 개발자에 의해서 소프트웨어 개발 전체가 유지되는 소프트웨어는 거의 존재하지 않는다.**

**③ 코드 규약은 가독성을 향상시켜 소프트웨어의 개발자들이 새로운 코드를 더 빠르고**

**완벽하게 이해할 수 있게 도움을 준다.**

**2. 개발환경**

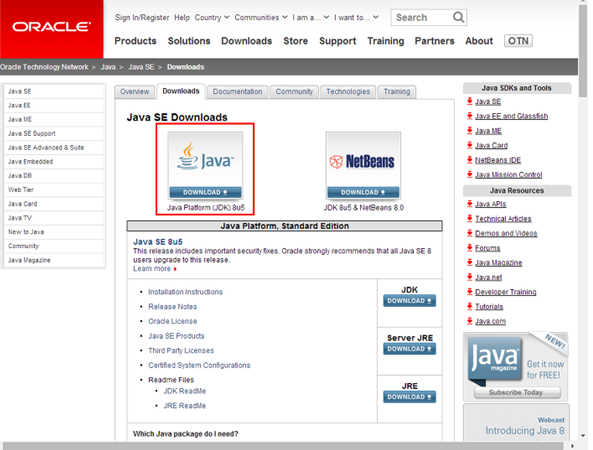
**2.1 JDK 1.7 version 설치 및 환경설정**

**2.1.1 JDK 설치**

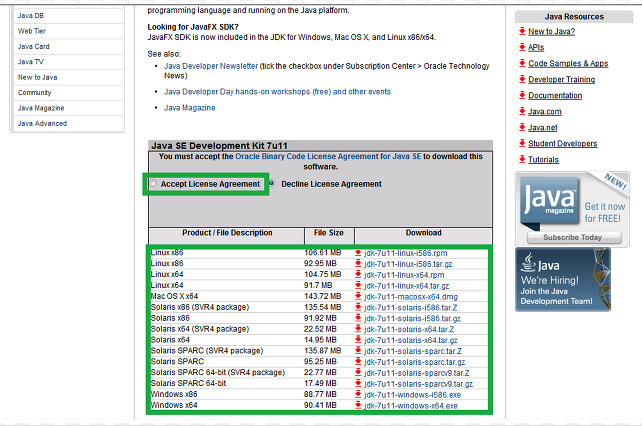
**아래 링크로 들어가서 Java Platform (JDK) 이것을 받습니다.**

1. **아래 링크로 들어가서 Java Platform (JDK) 이것을 받습니다.**

[**http://www.oracle.com/technetwork/java/javase/downloads/index.html?ssSourceSiteId=ocomen**](http://www.oracle.com/technetwork/java/javase/downloads/index.html?ssSourceSiteId=ocomen) **( 회사 공통 파일 서버 192.168.1.5\프로젝트\** **201410\_JF프로젝트\유틸리티\01.SpringTool\02.jdk 폴더에   
 jdk-7u67-windows-i586.exe 에서 받아 설치하면 된다.)**

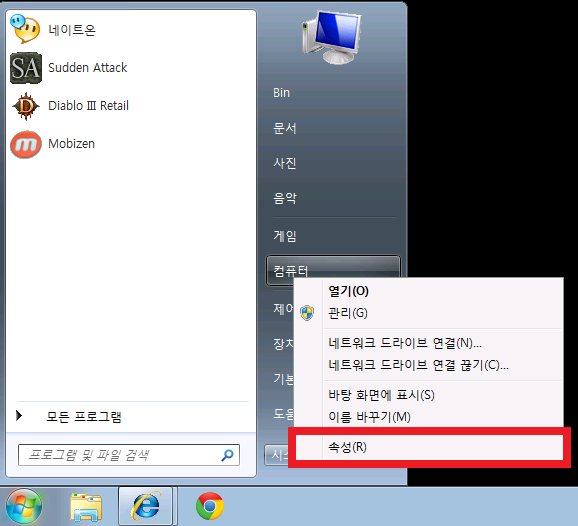


1. **Accept License Agreement(라이센스 동의)를 선택 후 아래의 목록에서 자신의 OS 버전과 일치하는 항목을 다운로드하고 설치를 합니다**



**2.1.2 JDK 환경 설정**

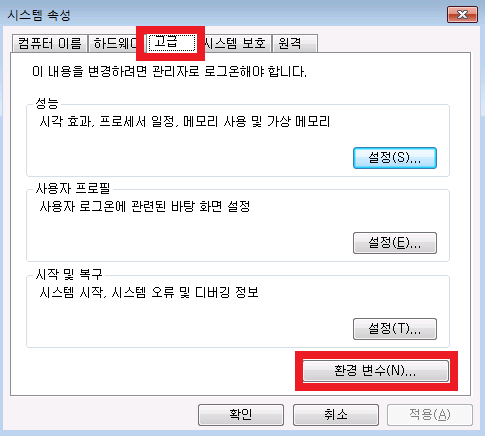
1. **내컴퓨터에서 마우스 오른쪽 클릭 후 속성을 선택합니다.**



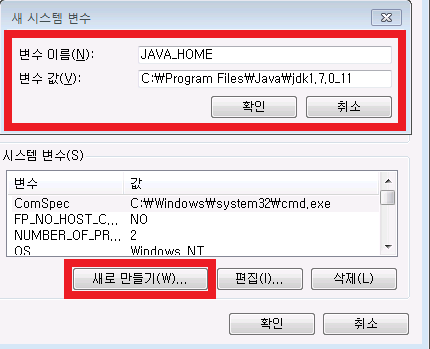
1. **고급 시스템 설정을 클릭합니다.**



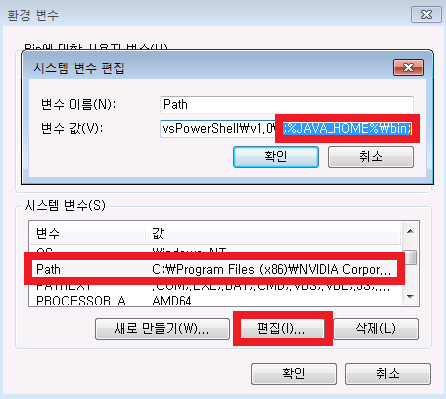
1. **고급 탭으로 이동 후 환경변수를 클릭합니다.**



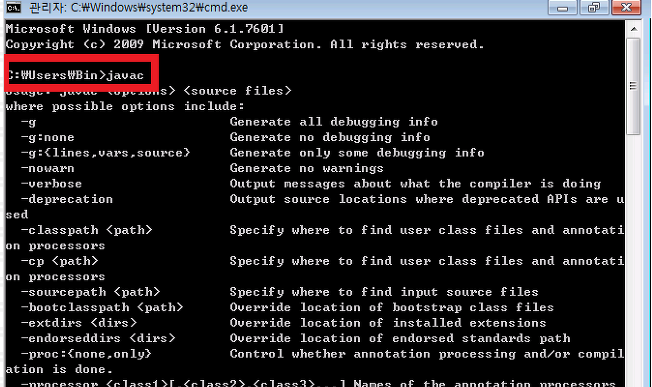
1. **새로 만들기 버튼을 클릭 후 변수이름은 JAVA\_HOME 변수 값은 JDK가 설치된 경로를 입력합니다. JDK 설치 시 설치폴더를 변경하지 않았다면 대부분 C:\Program Files\Java\jdk1.7.0\_11 일 것입니다. 위의 경로를 복사하여 사용 시 jdk1.7.0\_11에서 뒤의 버전은 본인이 다운로드한 버전으로 변경하시기바랍니다.**



1. **Path 변수는 이미 등록이 되어있을 것입니다. 만약 Path 변수가 없다면 위의 방법처럼 새로 만들기 후 추가하시면 됩니다. Path변수를 클릭 후 편집 버튼을 클릭합니다. 변수 값에는 이미 다른 값이 입력이 되어있을텐데, 그 뒤에 ;%JAVA\_HOME%\bin; 을 붙여줍니다. 주의할 점은 세미콜론입니다.**



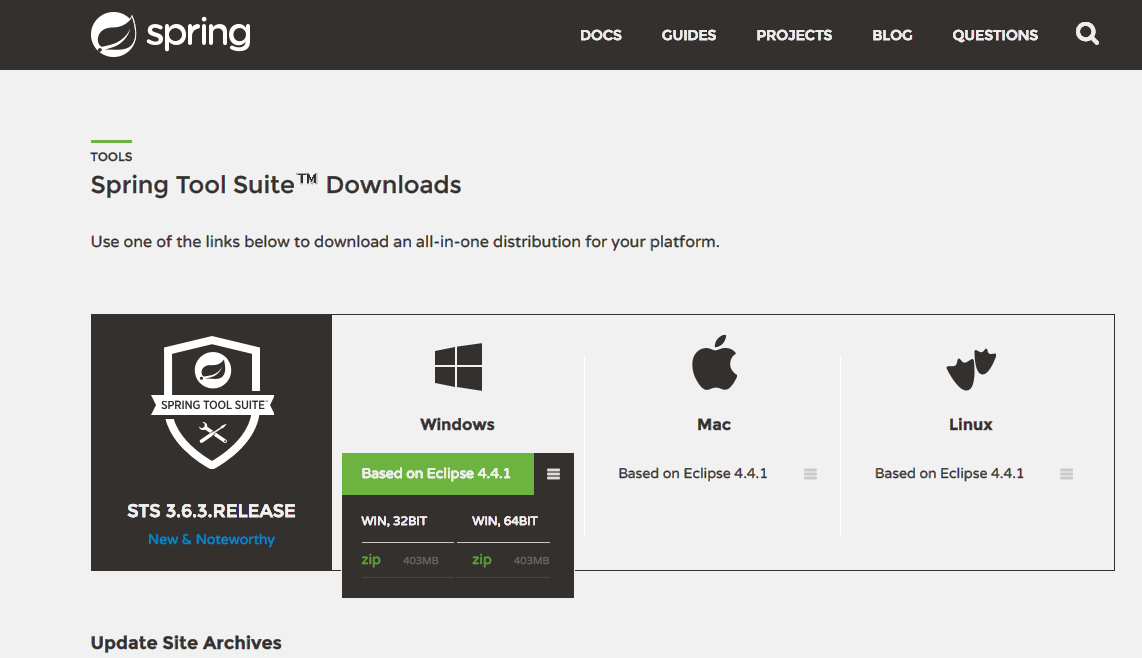
1. **Command창에서 javac라고 입력 후 엔터를 치면 아래와 같이 옵션 값들이 나오면 정상적으로 설치되었다는 증거입니다.**



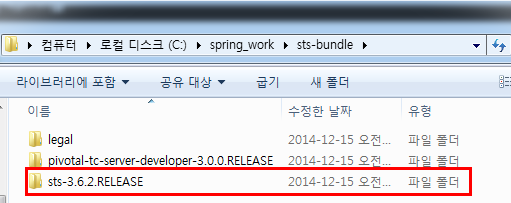
**2.2 Spring STS 설치**

* + 1. **STS(Spring Tool Suite) 설치**

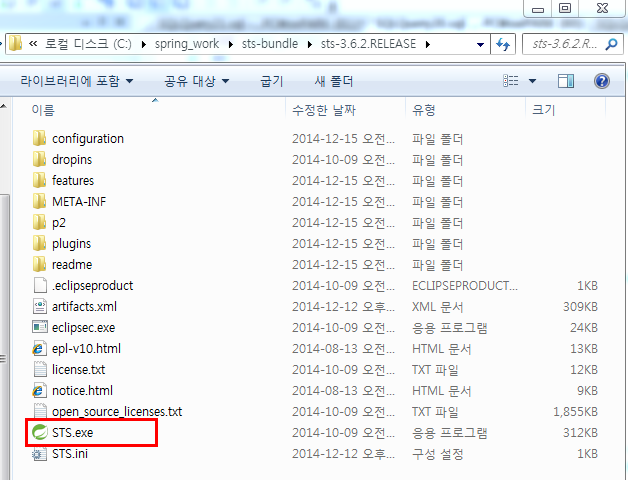
1. **http://spring.io/tools/sts/all 아래 경로로 들어가면 다운로드 파일이 사용하시는 운영체제의 bit에 맞춰 선택하여 다운로드 받는다. (회사 공통 파일 서버 192.168.1.5\프로젝트\** **201410\_JF프로젝트\유틸리티\01.Spring Tool\01.STS\spring-tool-suite-3.6.2.RELEASE-e4.4.1-win32.zip 설치 자료가 있다.)**



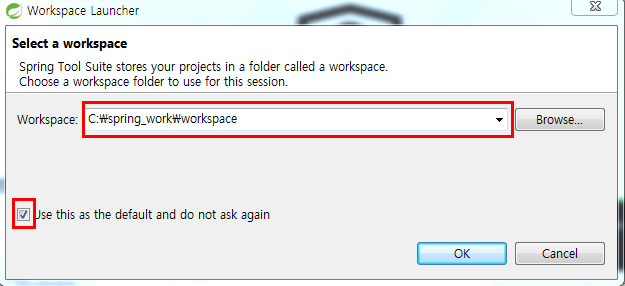
1. **원하는 경로를 만든다. 예 c:\spring\_work라는 폴더를 만들고 그 안에 다운 받은 STS 파일 압축을 푼다.**



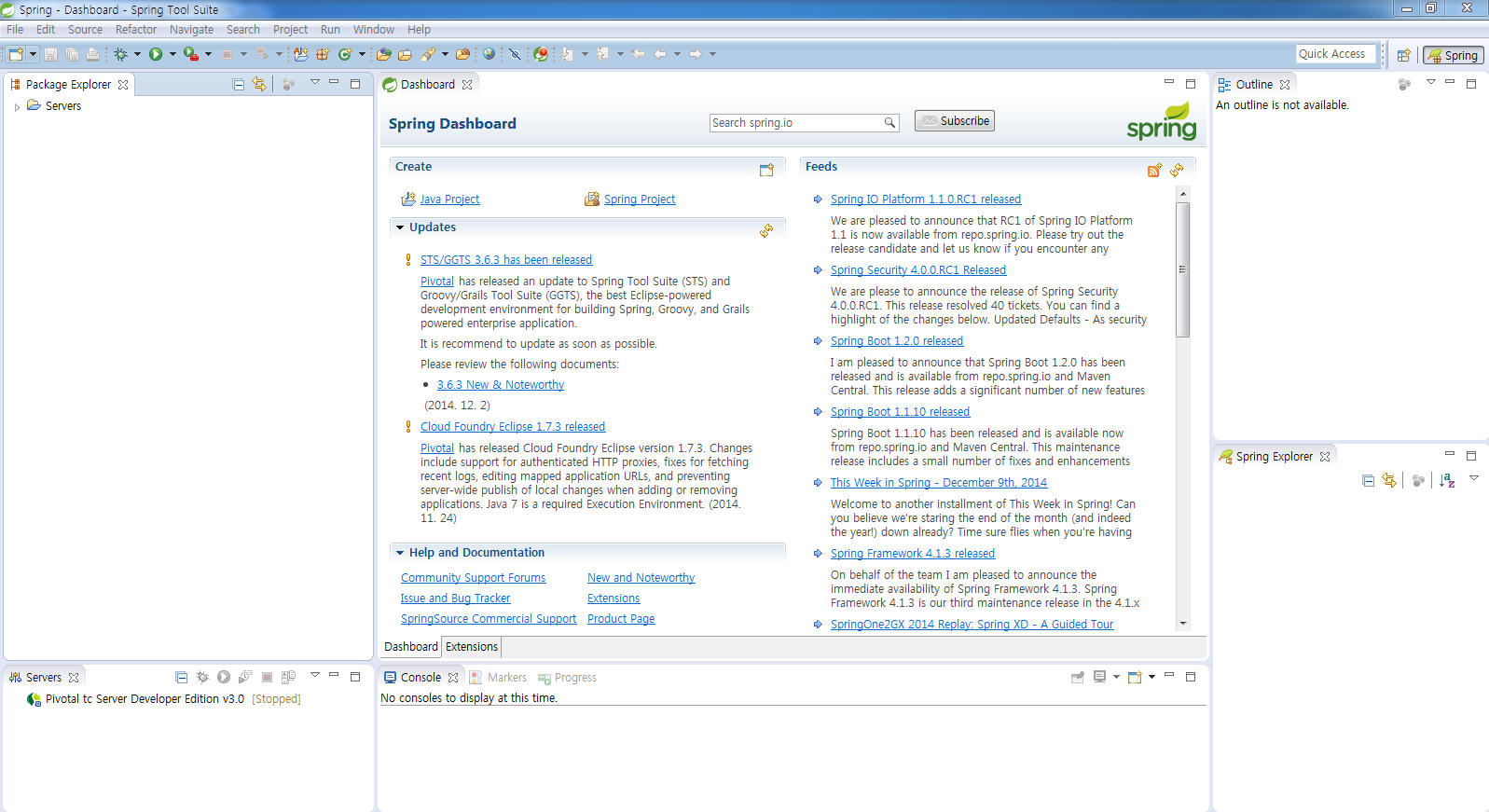
1. **실행 파일이 선택하여 실행한다.**



**처음 실행 시 Select a workspace 표시되면 (예 c:\spring\_work\workspace 폴더 생성) 미리 생성한 workspace 폴더를 선택한다.**



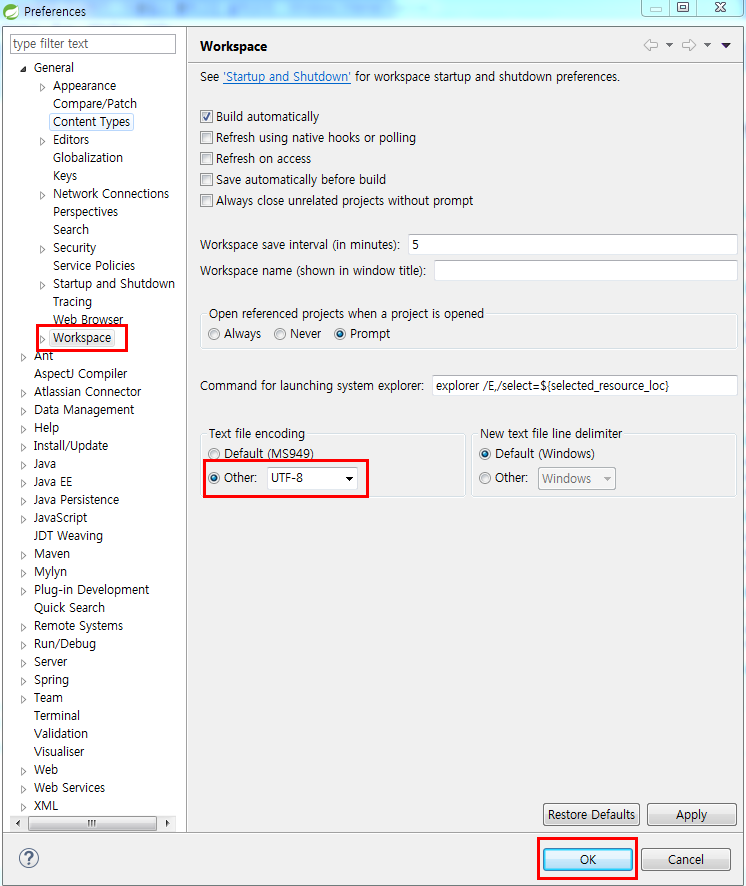
1. **STS 실행한 화면이 표시 되면 설치가 완료 되었다.**



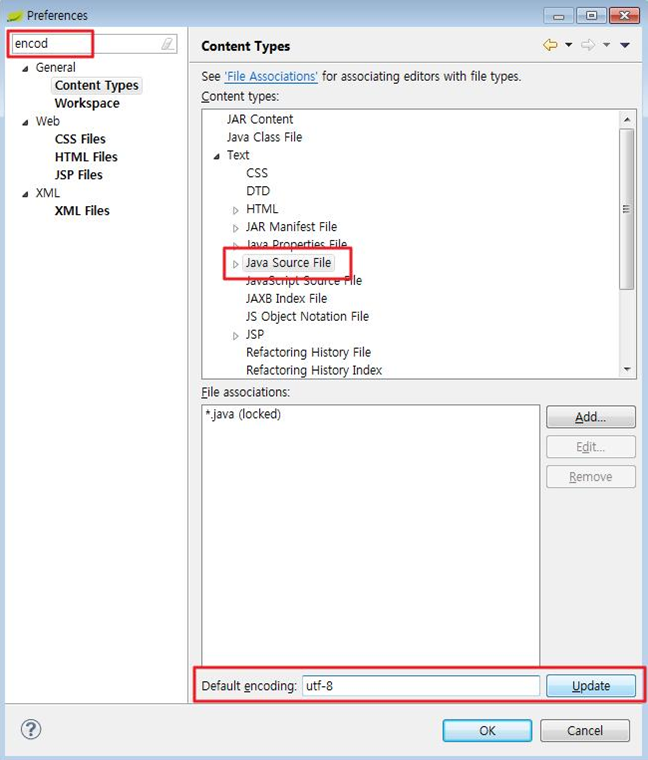
**2.3 환경 설정**

**2.3.1 STS(Spring Tool Suite) 언어코드 지정**

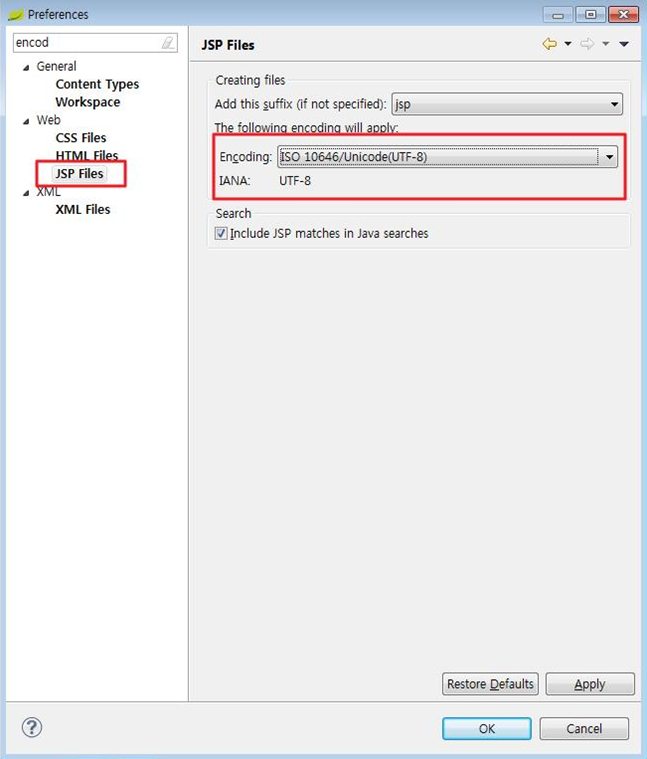
1. **Window->Preferences 선택 후 General->Workspace 선택하고 Text file encoding에서 Other 분에서 UTF-8로 변경 후 OK 버튼을 선택한다.**



1. **좌측 상단의 필터란에 encod 를 입력하여 나오는 항목 중 Content Types 에 Java Source 파일을 선택하여 utf-8 로 인코딩 타입을 지정해줍니다.**

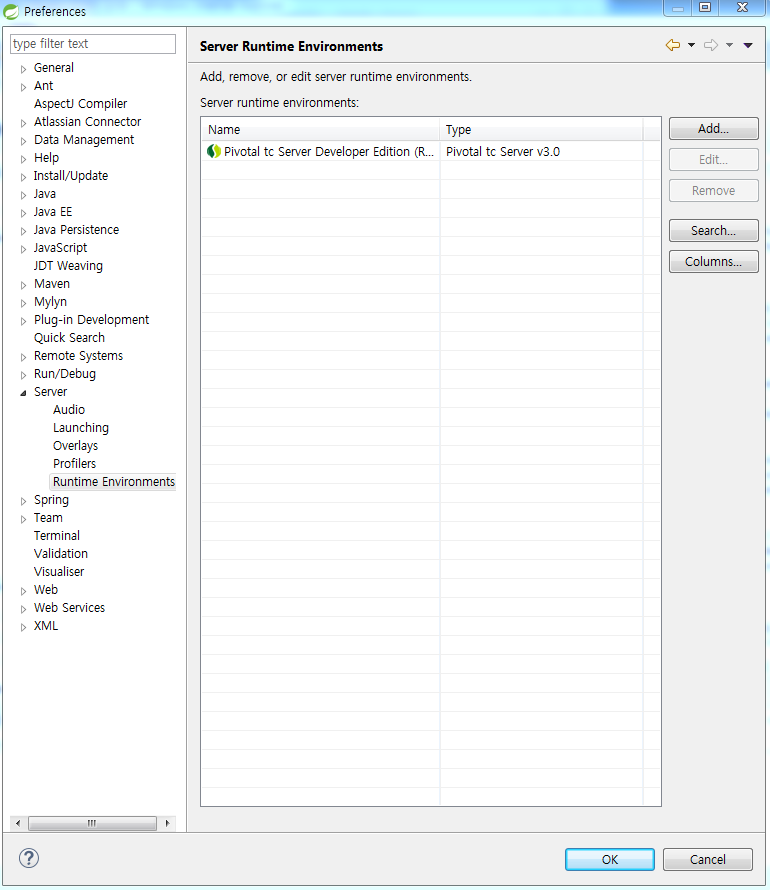


1. **xml 파일이나 jsp 파일을 선택하여 utf-8 방식으로 지정해줍니다.**

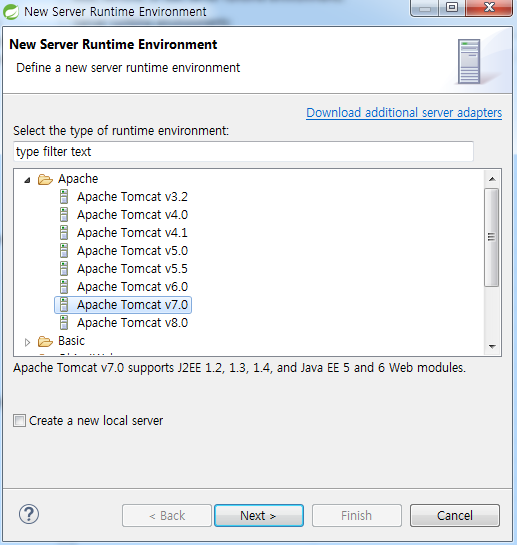


**2.3.2 Tomcat 설정**

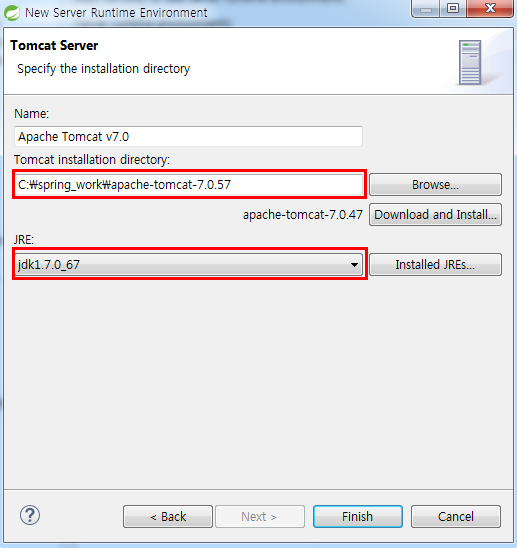
1. **상단의 Window > Preferences를 선택한다.**
2. **Server > Runtime Environment을 선택한다.**



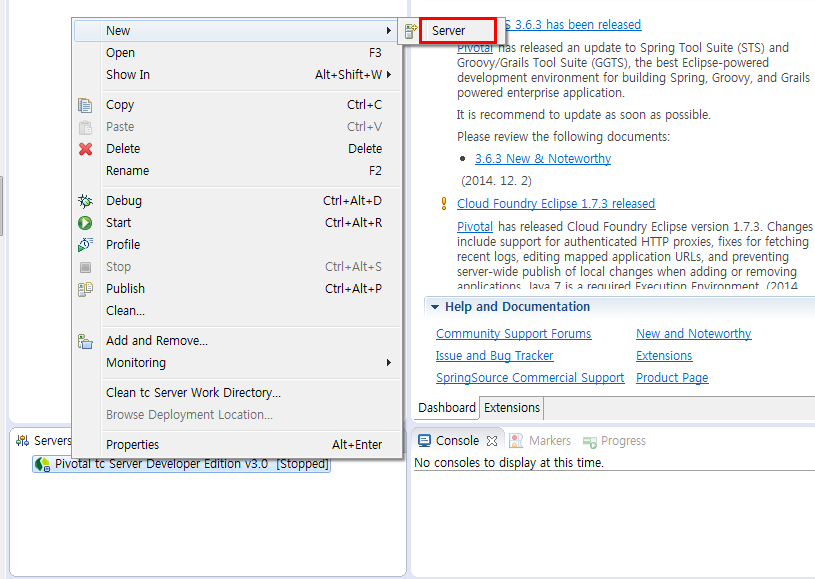
1. **Add 버튼을 누른다.**
2. **Apache tomcat 7.0을 선택하고 "Next >" 를 누른다.**



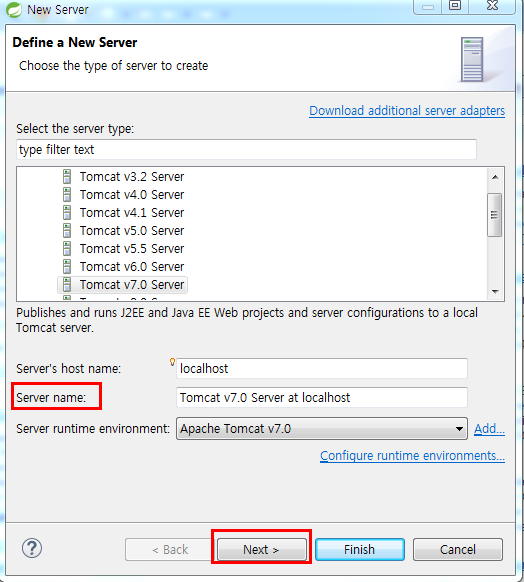
1. **JRE 및 tomcat 설정을 한다. c:\spring\_work 밑에 회사 공통 파일 서버 192.168.1.5\프로젝트\** **201410\_JF프로젝트\유틸리티\01.SpringTool\01.tomcat에서 apache-tomcat-7.0.57.zip 받아 압축을 푼 폴더를 선택하고 설치된 JRE를 선택하여준다. (c:\spring\_work는 STS와 workspace 지정한 작업 폴더로 임의로 개인 설정에 맞추어 변경이 가능하다)**



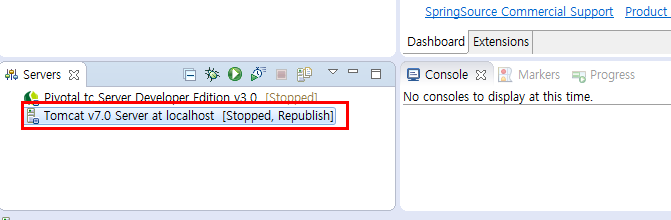
1. **Servers 탭에서 마우스 우클릭 후 New-> Server를 선택한다.**



1. **위에서 설치하셨던 tomcat 버전을 그대로 클릭하시면 Server Name에 위에서 등록했던 서버이름이 표시되면 Finish 버튼을 선택 후 설정을 마무리 한다.**

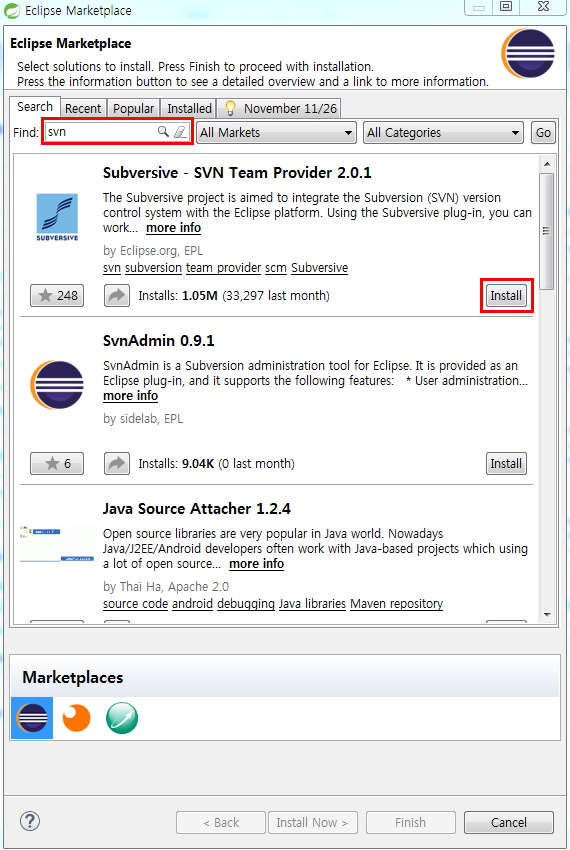


1. **Servers 탭에 tomcat 설치된 것을 확인 한다.**

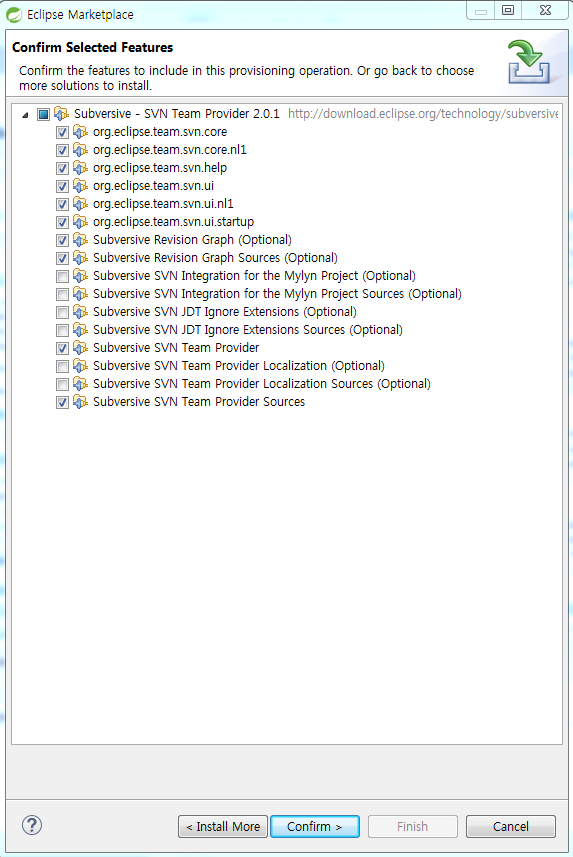


**2.3.3 SVN(형상관리 서버) 설정**

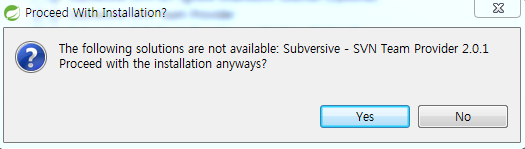
1. **상단의 Help > Eclipse Marketplace를 선택, find에 SVN을 입력하고 조회를 하고 Subversive- SVN Team Provider 2.0.1을 Install한다.**



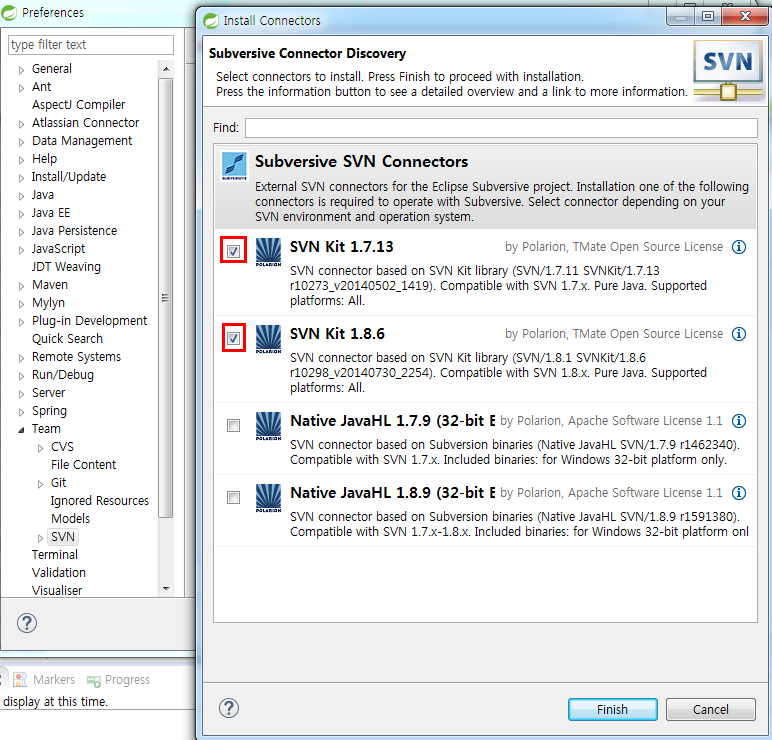
1. **아래 화면 처럼 필요한 Subversive -SVN Team Provider 2.0을 설치한다**

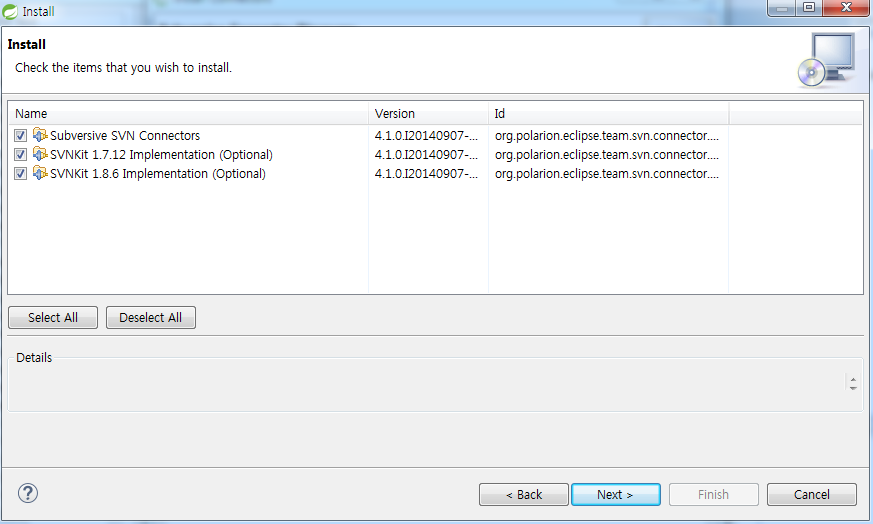


1. **라이센스 동의하고 설치를 진행한다**

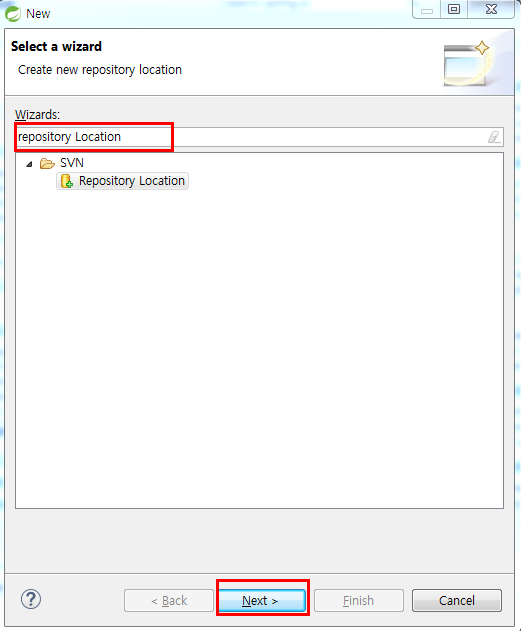


1. **이클립스를 재시작 할거냐고 물어보면 재시작을 한다.**
2. **Window->Preferences->SVN을 선택을 하면 Install connectors를 설치하라는 화면 표시되면 아래 그림 처럼 선택을 하고 Finish 버튼을 선택하여 설치 작업을 한다.**

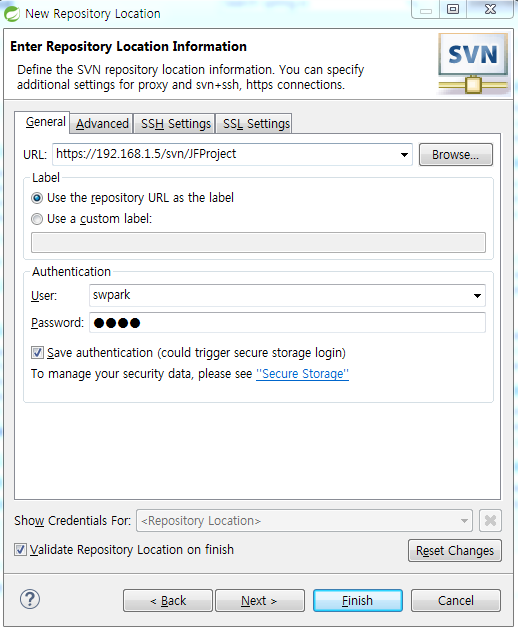




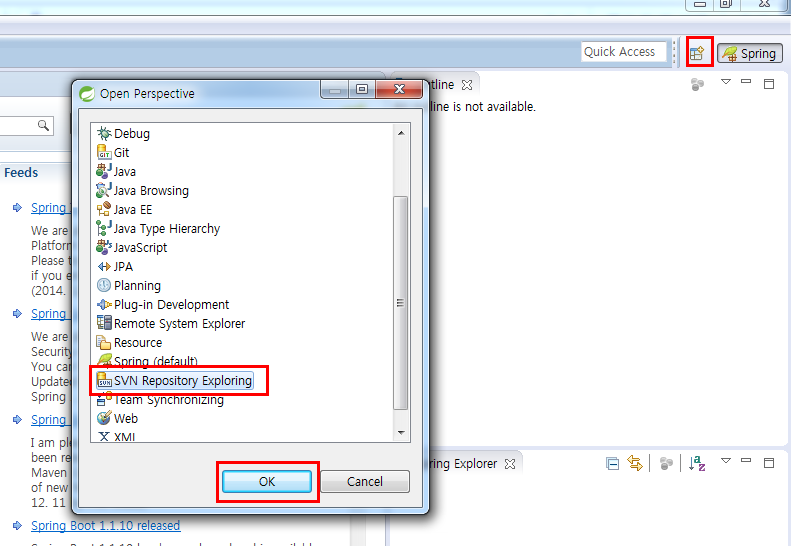
1. **Package Explorer에서 마우스 오른쪽 버튼을 선택하고 New->Other를 선택한 창에서 repository Location을 입력하고 표시되는 Repository Location을 선택하고 "Next >" 를 누른다.**



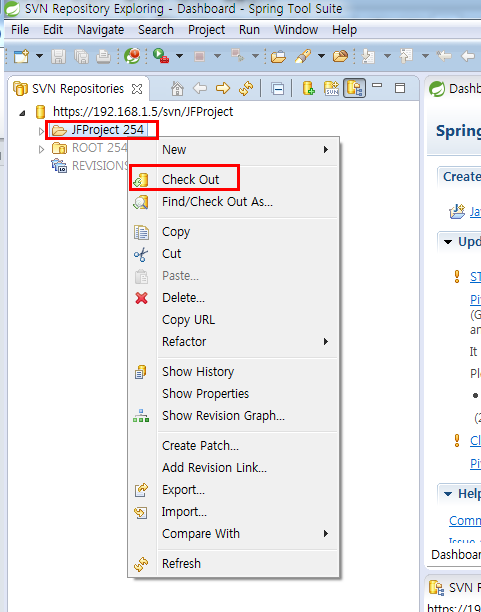
1. **Repository Location URL 부분에** [**https://192.168.1.5/svn/JFProject**](https://192.168.1.5/svn/JFProject)**을 입력하고 각 사용자에게 부여된 User ID, Password를 입력하고 Finish 버튼을 선택한다.**



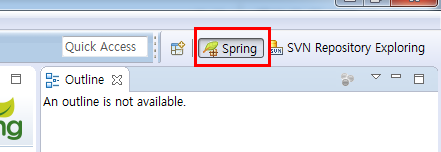
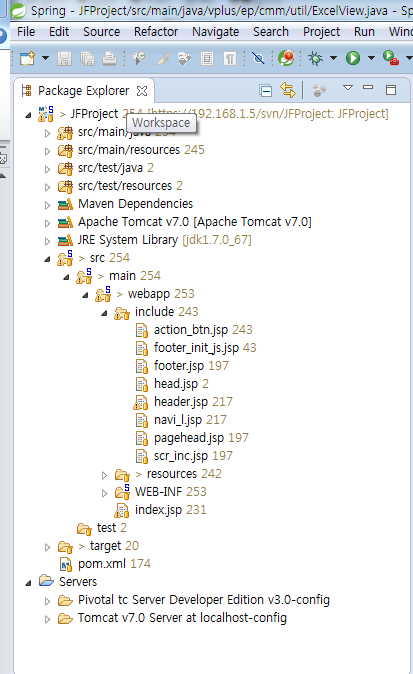
1. **Open Perspective 추가 화면에서 SVN Repository Exploring을 추가한다.**



1. **SVN Repository Exploring에서 다운 받을 Project를 선택하고 마우스 오른쪽 버튼을 클릭하고 Check out를 선택을 하여 표준 프로젝트를 다운 받는다.**

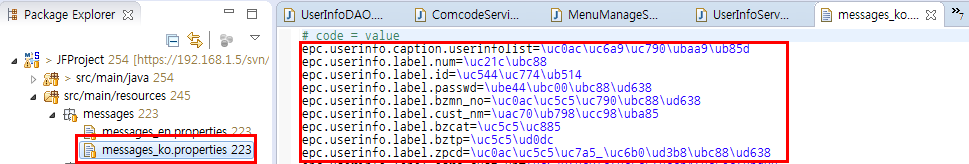


1. **Perspective를 Spring으로 변경을 하고 Package Explorer에 다운 받은 프로젝트가 존재하는지 확인을 한다.**

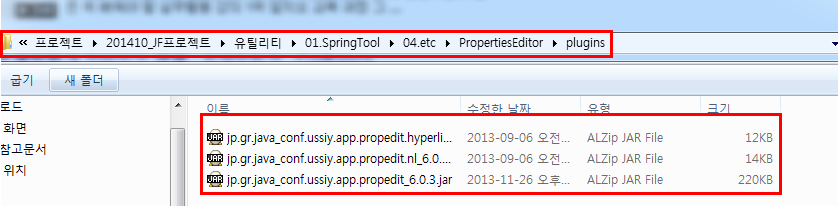
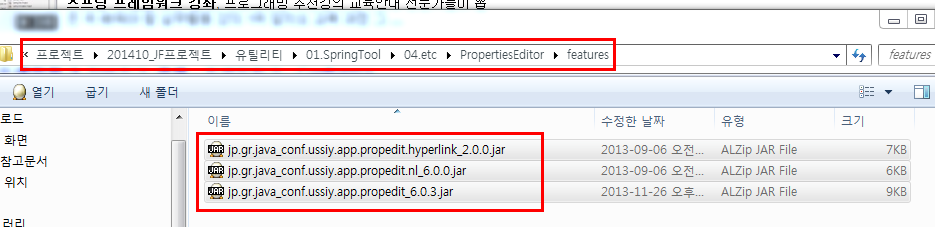


**2.3.4 properties editor 플러그인 설치**

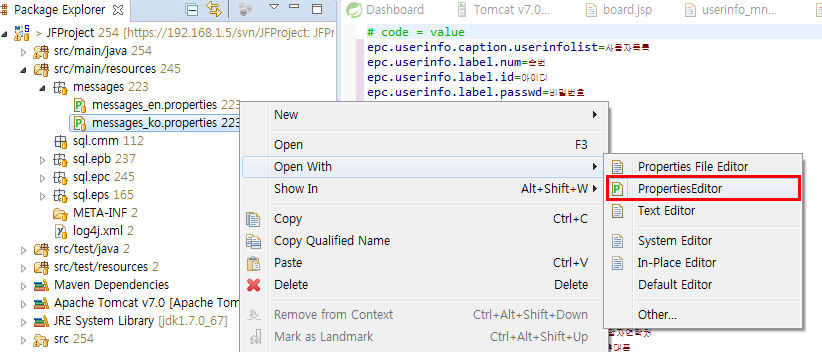
1. **한글 Message Properties 파일 자료 편집을 하기 위해 PropertiesEditor 플러그인이 설치한다. (Properties Editor를 설치 하지 않을 경우 한글 문자가 깨진 문자열 상태로 표시됨)**



1. **아래 그림에 파일들을 회사 공통 파일 서버 192.168.1.5\프로젝트\ 201410\_JF프로젝트\유틸리티\04.etc \Properties Editor\plugins 와 \PropertiesEditor\features 에서 복사하여 위에서 설치한 STS 프로그램 위치 C:\spring\_work\sts-bundle\sts-3.6.2.RELEASE\plugins 과 \feactures 폴더에 복사 한다.**



1. **만약 이클립스가 사용 중이면 재시작을 하고 재시작 후 아래 그림처럼 messages properties 파일을 선택하고 open With에서 PrpertiesEditor를 선택한다.**



**3. 분류 코드**

**3.1 시스템 코드 (ex) B2B**

**① 모듈코드는 영문 소문자로 표기한다.**

**② 표현형식 : ‘ep’ + 모듈의 첫 문자**

|  |  |  |  |
| --- | --- | --- | --- |
| **모듈명** | **모듈코드** | **의미** | **패키지** |
| **공통** | **cmm** | **common util** | **vplus.ep.cmm** |
| **커뮤니티** | **epb** | **board** | **vplus.ep.epb** |
| **공통관리** | **epc** | **common** | **vplus.ep.epc** |
| **시스템관리** | **eps** | **system** | **vplus.ep.eps** |

**4. 디렉토리 및 파일**

**4.1 프로그램 및 업무 속성에 따른 디렉토리의 분리**

**프로그램 및 업무 속성에 따라 프로그램 및 업무 속성을 반영하여 디렉토리를 설정하는 것으로 한다.  
Spring Framework Based Framework 기본 폴더의 구성은 아래와 같이한다.**

**① Class File 경로**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **기본경로** | **하위경로** | | | **비고** |
| **JFProject /src /main /java** | **/모듈코드** | **/ctrl** |  | **Controller** |
|  |  | **/model** | **/dao** | **Db Access Object** |
|  |  |  | **/vo** | **Value Object** |
|  |  | **/service** | **/impl** | **Service** |
| **JFProject /src /main /resources** | **/messages** |  |  | **locale Message** |
|  | **/sql** | **/모듈코드** |  | **SQL Xml** |

**② View File 경로**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **기본경로** | **하위경로** | | | **비고** |
| **JFProject /src /main /wepapp** | **/include** |  |  | **head, header ...** |
|  | **/resources** | **/css** |  | **CSS** |
|  |  | **/images** |  | **Image** |
|  |  | **/js** |  | **Board SmartEdit** |
|  |  | **/smartEdit** |  |  |
|  |  |  |  |  |
|  | **/WEB-INF** | **/spring** |  | **context XML** |
|  |  | **/views** | **/ep /모듈코드** | **JSP** |

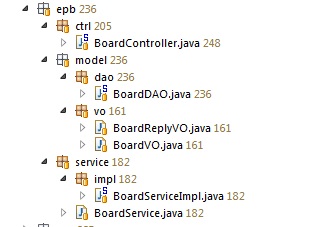
**5. 자바 코딩 표준**

**5.1 자바 소스 파일**

**5.1.1 파일구조**

|  |  |  |
| --- | --- | --- |
| **항목** | **파일명** | **프로그램 명** |
| **Controller** | **프로그램 명 + Controller.java** | **- 대문자로 시작하는 명사형 단어**  **- 복합단어는 각 단어의 첫 글자가 대문자**  **ex) BoardListController.java** |
| **Service** | **프로그램 명 + Service.java** |
| **Service** **Implements** | **프로그램 명 + ServiceImpl.java** |
| **VO** | **프로그램 명 + VO.java** |
| **DAO** | **프로그램 명 + DAO.java** |
| **SQL Map** | **프로그램 명 + .xml** | **- 소문자의 명사형단어**  **ex) boadlist.xml** |

**( 5.1.1 Package Example )**

****

**5.1.2 명명 규칙**

**명명 규칙은 프로그램을 이해하기 쉽게 만들어 주며 식별자의 기능에 대한 정보를 제공한다. 예를 들어, 그것이 상수인지 패키지인지 클래스인지를 알게 해 준다. 이러한 정보는 코드를 이해하는데 도움을 준다.**

**① 타입별 명명 규칙**

|  |  |  |
| --- | --- | --- |
| **식별 타입** | **명명 규칙** | **예제** |
| **패키지** | **vplus.ep.모듈코드.클래스타입 으로 사용한다.** | **package vplus.ep.eps.ctrl** |
| **클래스** | **클래스명은 명사로 작성하며, 복합단어는  각 단어의 첫 글자를 대문자로 작성한다.**  **클래스 이름 생성시 파일명을 따라가며  명명Controller, 명명Service, 명명ServiceImpl,  명명DAO, 명명VO 와 같은 형태로 작성한다.** | **class UserInfoController**  > **class UserInfoService**  > **class UserInfoServiceImpl**  > **class UserInfoDAO**  > **class UserInfoVO** |
| **메소드** | **메소드명은 동사로 작성하며, 복합단어는  첫 글자를 소문자로 작성하고 각 단어의 첫 글자는 대문자로 작성한다.  메소드 타입에 따라 접두어와 under score 를 사용하며, 세부사항은 별도 첨부된 메소드 명명표를 참고한다.** | **public String create\_UserInfo** |
| **변수** | **변수 명은 소문자로 작성하여 복합단어는 under score 로 연결한다. 첫 글자를 소문자로 작성하고 각 단어의 첫 글자는 대문자로 작성한다.** | **String result\_Msg** |

**② 타입별 메소드 명명 규칙**

|  |  |  |  |
| --- | --- | --- | --- |
| **메소드 타입** | **접두어** | **비고** | **예제** |
| **생성** | **create\_ (or) add\_** | **데이터 생성 혹은 추가** | **public String create\_UserInfo** |
| **검색** | **get\_ (or) check\_** | **데이터 검색 혹은 검증** | **public String get\_UserInfo** |
| **수정** | **modify (or) change\_** | **데이터 수정 혹은 변경** | **public void modify\_UserInfo** |
| **제거** | **remove\_** | **데이터 제거** | **public void delete\_UserInfo** |
| **Form Type** | **mng\_** | **Form Type 에 따른 CRUD 사용** | **public String mng\_UserInfo** |
| **Page Move** | **move\_** | **Request Mapping 후 페이지 이동** | **public String move\_UserInfo** |

**③ 타입별 VO (Value Object) 명명 규칙** [ 5.1.3 주석 ④ VO 주석 그림 참조 ]

|  |  |  |
| --- | --- | --- |
| **VO 타입** | **명명 규칙** | **예제** |
| **컬럼** | **Data Connection 을 위해 DB 컬럼명을 그대로 사용한다.** | **private String RMKS;**  **private String TEL\_NO;** |
| **변수** | **소문자로 작성하여 복합단어는 under score 로 연결한다.**  **첫 글자를 소문자로 작성하고  각 단어의 첫 글자는 대문자로 작성한다.** | **private String password private String login\_Result** |

**5.1.3 주석**

**① Controller : 기본 주석을 상세히 기술한다.**

|  |
| --- |
| **/\***  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\* 시스템명 : B2B**  **\* 프로그램ID : LoginController**  **\* 프로그램명 : 로그인 페이지**  **\* 작성자 : 홍순모**  **\* 최초작성일 : 2014-11-12**  **\* 버전 : v1.0**  **\* 변경이력**  **\* 1. 최초작성**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*/**  package vplus.ep.eps.ctrl; |

**② Service, ServiceImpl, DAO : 약식 주석을 간략히 기술한다.**

|  |
| --- |
| **/\***  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\* 프로그램ID : LoginService**  **\* 작성자 : 홍순모**  **\* 최초작성일 : 2014-11-12**  **\* 변경이력**  **\* 1. 최초작성**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*/**  package vplus.ep.eps.service; |

**③ 각 메소드는 JavaDoc 주석** ( /\*\* 입력 시 자동생성 ) **을 추가한 뒤 필요내역만 기재한다.**

|  |
| --- |
| **/\*\***  **\* 로그인체크**  **\*/**  ***@param @return @throws 제거***  public Account Get\_UserLogin( String id, String pswd ) throws Exception; |

**④ VO : 약식 주석을 간략히 기술하고, DB Table VO는 상단에 테이블 정보, 우측에 Colum 정보를 기재한다**

|  |
| --- |
| **/\***  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\* 프로그램ID : MenuManage**  **\* 작성자 : 홍순모**  **\* 최초작성일 : 2014-11-12**  **\* 변경이력**  **\* 1. 최초작성**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*/**  package vplus.ep.epc.model.vo;  public class MenuManageVO {    **/\*\***  **\* Table Name : TEMZ\_PROGCODE**  **\*/**  **private** String SEQPROG; **// 자동순번 (key)**  **private** String PROG\_NM; **// 프로그램명**  **private** String PROG\_EXEC\_CD; **// 실행코드**  **private** String PROG\_LANG\_NM; **// 프로그램명\_lANG**  **private** String RMKS; **// 비고**  **private** String IPT\_ID; **// 입력자**  **private** String IPT\_DT; **// 입력일시**  **private** String UPD\_ID; **// 수정자**  **private** String UPD\_DT; **// 수정일시**    **// 변수**  **private** String formType;  **private** String sch\_Type;  **private** String sch\_Text;  ; |

**5.1.4 클래스**

**① Controller / Service / DAO 작성 기준**

|  |  |
| --- | --- |
| **클래스 타입** | **작성 기준** |
| **Controller [Data 전달]** | **view단에서 넘어온 데이터 유효성 검사,  Service로 처리요청 후 이를 리턴받아 서버측 처리결과를 view단으로 리턴** |
| **Service [Data 가공]** | **비즈니스 로직을 수행 DAO를 통해 data를 호출하여 Controller 로부터의 처리요청을 수행** |
| **DAO [Data 호출]** | **DB Access 만을 담당하며 비즈니스 로직은 기재하지 않는다. 단일 데이터를 접근, 갱신을 처리하며 다중 DB 접근을 지양한다. parameter VO 처리는 Service 에서 처리한다.** |

**② Request Mapping**

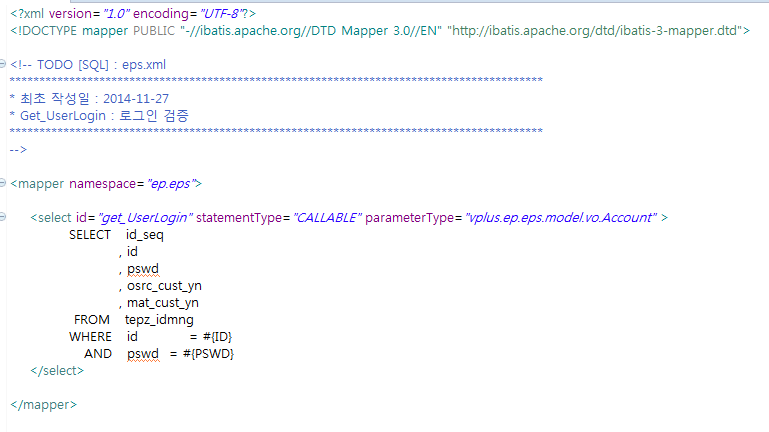
**- A. 호출 URL의 모듈코드까지를 단위 클래스에 매핑   
- B. 상세 호출 URL을 해당 메소드에 매핑하여 기재한다.**

|  |
| --- |
| **package** vplus.ep.epc.ctrl;  @Controller  **@RequestMapping("/ep/epc") “/ep/모듈코드” 를 클래스에 매핑.**  **public** **class** UserInfoController {  @**RequestMapping(value = "/userinfo\_mng.do")**  **“valule = 상세 호출경로” 메소드에 매핑**  **public** String userInfoMgm(HttpServletRequest request,  HttpServletResponse response, Model model) **throws** Exception {  **return** "/ep/epc/userinfo\_mng";  }; |

**6. SQL**

**6.1 SQL Map**

**① 기본 주석을 작성하고, 참조하는 Service 정보를 추가로 기술한다.**

****

**② mapper namespace 는 참조하는 dao 상위 패키지 명을 사용하여 id 중복을 방지한다. (+프로그램명 추가 고려)**

**③ id 는 ‘접두어\_ + 쿼리명’ 의 형태로 작성하며 첫 글자는 소문자 각 단어의 첫 글자는 대문자로 작성한다.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ansi Query** | **ID** | **Stored Procedure** | **ID** |
| **Insert** | **ins\_ + 쿼리명** | **실행 쿼리** | **exe\_ + 쿼리명** |
| **Select** | **get\_ + 쿼리명** | **리턴 쿼리** | **get\_ + 쿼리명** |
| **Update** | **upt\_ + 쿼리명** |  |  |
| **Delete** | **del\_ + 쿼리명** |  |  |

**④ 쿼리 작성시 컬럼 조건문은 한 줄씩 작성하며, 예약어와 파라미터는 대문자 이외의 문자는 소문자로 작성한다.**

**⑤ 예약어는 오른쪽 정렬, 컬럼은 왼쪽정렬은 사용한다.**

**⑥ Select 문은 작성시 최상위 문장은 한 줄에 한 컬럼씩 작성한다.**

**⑦ Colum 앞에 ‘,’ 를 기재하고 공백 1칸을 넣는다.**

**⑧ Sub Query 의 경우 한 줄에 작성하며, 컬럼의 길이가 긴 경우 2~5개씩 맞추어 사용한다.**

**⑨ From 절의 테이블명은 1줄당 1개씩 작성하는 것을 기본으로 하며, 테이블이 많은 경우 2개씩 작성한다.**

**⑩ Where 절의 조건은 1줄당 1개의 조건을 작성한다.**

**6.2 Mybatis**

|  |  |
| --- | --- |
| **datasSource**  **: Database Connect 설정** | |
| **driverClassName** | **MS SQL Server Driver** |
| **url** | **Database Url** |
| **username / password** | **Database ID/PW** |
| **sqlSessionFactory**  **: Database Connect 설정** | |
| **sqlSessionFactory** | **Project 내 SqlMap 경로 지정** |

**( 5.1.6 root-context.xml Example )**

****

**7. 화면디자인**

**7.1 화면 소스파일**

**7.1.1 파일구조**

**- 프로그램명은 소문자의 명사형 단어로 지정하며 복합단어는 단어 사이에 ‘\_’를 넣어 연결 ex) board\_action.jsp**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **기본경로** | **하위경로** | | | **비고** |
| **JFProject /src /main /wepapp** | **/include** |  |  | **head, header ...** |
|  | **/resources** | **/css** |  | **CSS** |
|  |  | **/images** |  | **Image** |
|  |  | **/js** |  | **Board SmartEdit** |
|  |  | **/smartEdit** |  |  |
|  |  |  |  |  |
|  | **/WEB-INF** | **/spring** |  | **context XML** |
|  |  | **/views** | **/ep /모듈코드** | **JSP** |

**( 7.1.1 Package Example )**

****

**7.1.2 View 코드 명명규칙**

|  |  |  |
| --- | --- | --- |
| **요소** | **규칙** | **명칭** |
| **Button** | **명칭은 btn을 접두어로 사용**  **접두어 뒤에는 특성을 대표할 수 있는 단어를 사용** | **btnAddRow, btn** |
| **Form** | **form의 기본 명칭은 form1**  **검색을 위한 form의 기본명칭은 formS1**  **중복된 것은 명칭의 1을 다른 숫자로 붙임**  **(숫자는 순서대로 증가)** | **form1, form2, …** |
| **Input** | **대문자**  **db column명과 같게**  **중복된 것은 명칭 뒤에 숫자를 붙임** | **DTL\_CD, USE\_YN, …** |
| **Select / Option** |
| **jqGridListBox** | **jqGrid의 기본 명칭은 jqGridList1**  **jqGrid가 중복됬을 경우에는 명칭의 1을 다른 숫자로 붙임**  **(숫자는 순서대로 증가)** | **jqGridList1Box, jqGridList2Box, …** |
| **jqGridList** | **jqGridList1, jqGridList2, …** |
| **jqGridListPage** | **jqGridList1Page, jqGridList2Page, …** |

**7.1.3 주석**

**- 주석은 태그 라이브러리와 html 시작 사이에 아래와 같은 형식으로 넣어줌**

****

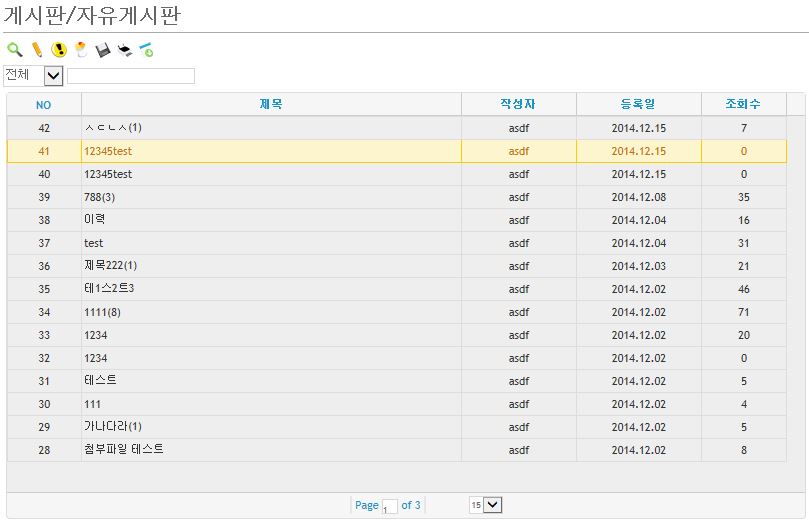
**7.2 View Pattern**

**4가지의 View Pattern이 있음**

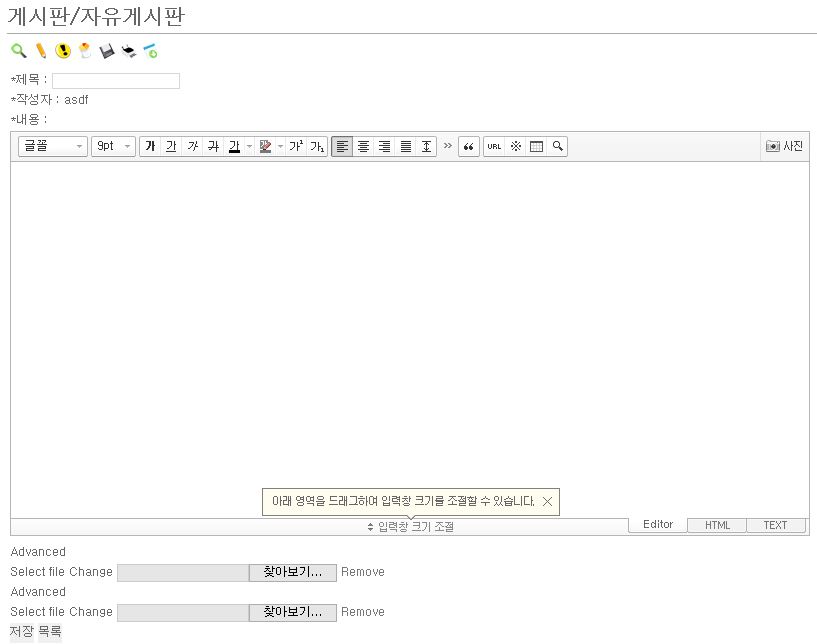
**① 1 Grid + Board Write**

**- 리스트를 출력**

**- 해당 row 클릭 시 페이지 이동되며 상세내용 출력**

****

**- smartEditor를 이용하여 게시글 및 첨부파일 저장**

****

**② 2 Grid**

**- 상위 그리드 row 클릭 시 하위그리드 출력**

**- 상위 그리드는 input 폼에 신규 입력하며 하위 그리드는 그리드 내에서 신규 입력, 수정**

**- 해당 row 더블클릭 시 수정가능**

****

**③ PopupGrid**

**- grid에서 해당 row를 더블클릭 하면 팝업창이 뜨며 그 팝업창 내에서 수정**

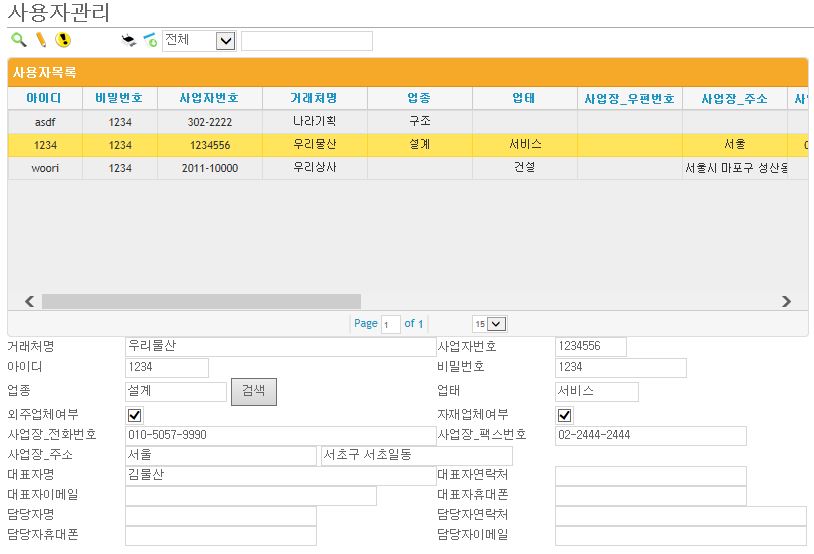
**- 신규 입력은 팝업창에서 하지 않으며 기존 그리드 내에서 입력**

****

**④ 1 grid + 1 form**

**- grid에서 해당 row를 클릭하면 하단 form에 뿌려짐**

**- 입력과 수정은 grid에서 하지 않으며 form내에서 함**

****

**7.3 View Pattern – 공통**

**7.3.1 HTML**

**① 태그 라이브러리를 선언**

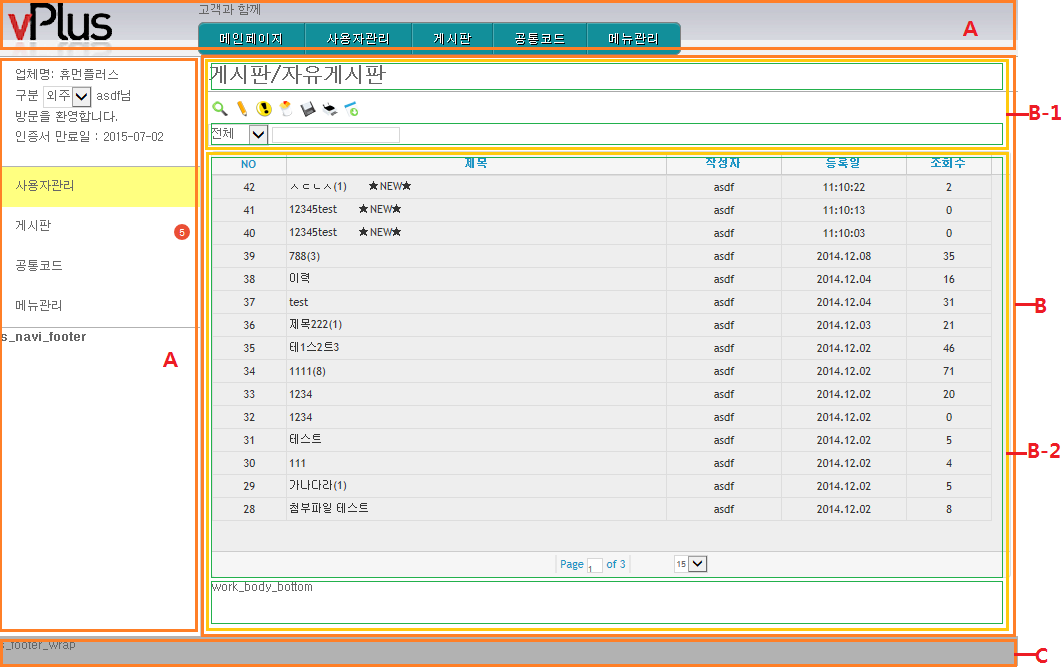
****

**② Head 부분 include**

**- 공통적으로 필요한 플러그인이 정의된 pagehead를 include**

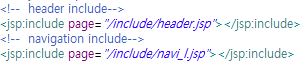
****

**③ Body**

****

|  |  |  |
| --- | --- | --- |
| **구분** | **구분 class** | **내용** |
| **A** | **P-HEADER-FIXED** | **상단 메뉴 관련 Header include**  **좌측에 위치한 사용자 정보와 메뉴 관련 navi\_l를 include** |
| **B** | **P-WORK-BODY** | **WORK BODY HEAD + WORK BODY** |
| **B-1** | **WORK-BODY-HEAD** | **메뉴명 / 상단 button관련 action\_btn를 include** |
| **B-2** | **WORK-BODY** | **jqGrid를 뿌려줄 테이블 생성**  **submit시 필요한 파라미터를 담을 form을 생성** |
| **C** | **FOOTER** | **footer** |

**- A(P-HEADER-FIXED)**

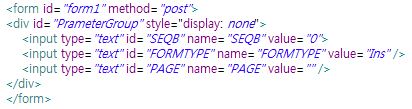
****

**- B-1(WOK-BODY-HEAD)**

****

**- B-2(WORK-BODY)**

****

****

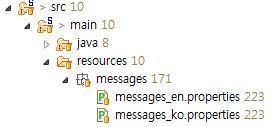
****

**- C(FOOTER)**

****

**④ 다국어 처리**

**- JFProject /src /main / resources / message 패키지 내에 messages\_en.properties와 messages\_ko.properties**

****

**- 아래와 같이 영문은 en.properties에 정의하고 한글은 ko.properties에 정의한다**

****

****

**- 사용하고자 하는 곳에**

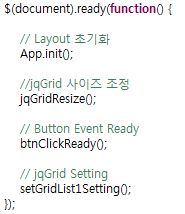
**<spring : message code = “messages\_en.properties와 messages\_ko.properties에 정의해 둔 코드명” />**

****

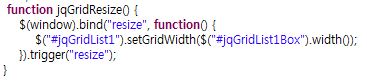
**7.3.2 Script**

**① 문서를 열자마자 제일 먼저 실행되는 부분**

**- layout init, jqGrid Resize, button event, grid setting 호출**

****

**② jqGrid Resize**

****

**③ Button Event Ready**

**- 버튼 이벤트는 btnClickReady 내에 모두 정의**

****

**④ jqGrid Setting**

**- jqGrid Option**

|  |  |
| --- | --- |
| **Option** | **설명** |
| **url** | **데이터를 받아올 url(요청주소)** |
| **mtype** | **요청방식(GET/POST)** |
| **datatype** | **결과물로 받을 데이터 타입(json/xml)** |
| **caption** | **그리드 캡션** |
| **colNames** | **Header에 출력되는 각 column의 이름을 배열정보로 설정** |
| **height** | **그리드의 높이** |
| **autowidth** | **자동으로 column 가로 크기 조정(true/false)** |
| **rowNum** | **그리드(페이지)마다 보여줄 row의 수** |
| **rowList** | **row변경 옵션(유저 선택에 의해 화면에 출력할 row의 수)** |
| **sortname** | **초기 데이터 정렬 시 정렬 조건(columnName)** |
| **onSelectRow** | **Row 선택시에 호출** |
| **pager** | **페이지를 표현하는 도구모음이 화면에 보일 html의 id값** |
| **colModel** | **각 column에 대한 상세정보 설정** |
| **navGrid** | **하단 page표시 부분에 jqGrid가 제공하는 search / edit / add / del / page기능에 대한 사용 여부(true / false)** |

****

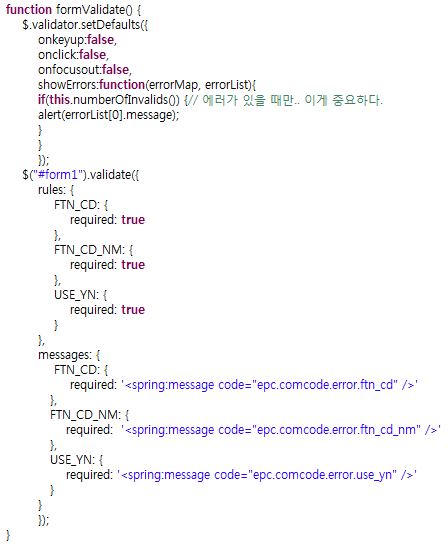
**⑤ Validation**

**- 입력/수정이 있는 form은 validation 체크 해야함**

**- rules에 required가 true로 지정된 요소는 필수 입력**

**- message 처리는 submit message처리와 동일하게**

**<spring : message code = “messages\_en.properties와 messages\_ko.properties에 정의해 둔 코드명 입력” />**

****

**⑥ ajaxSubmit**

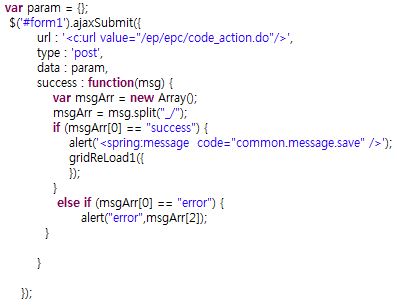
**submit(입력, 수정, 삭제)은 ajaxSubmit 사용**

|  |  |
| --- | --- |
| **option** | **설명** |
| **url** | **요청을 보낼 url** |
| **type** | **전송 타입** |
| **data** | **보낼 데이터** |
| **success** | **성공 후 처리하는 부분**  **(msg를 결과값으로 받아와 처리)** |

**- submit 후 message처리는**

**<spring : message code = “messages\_en.properties와 messages\_ko.properties에 정의해 둔 코드명 입력” />**

**message 처리부분에 save / update / delete 코드를 넣어줌**

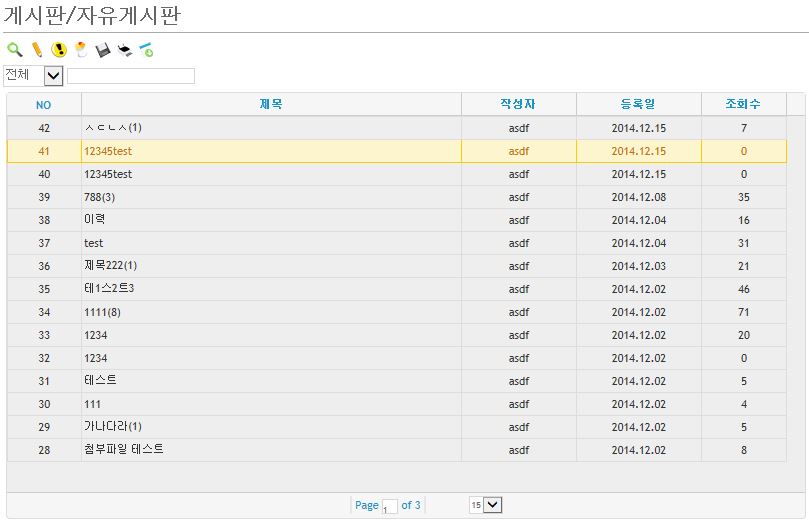
****

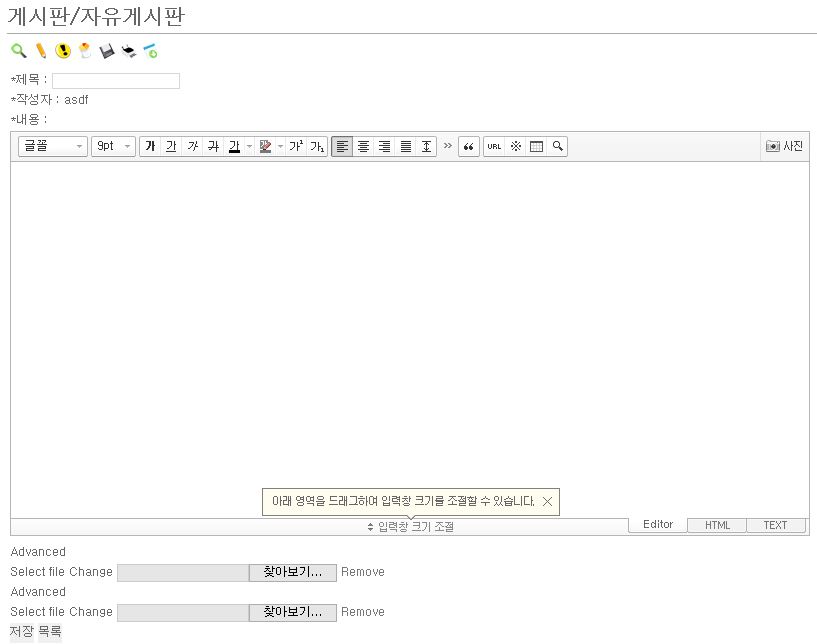
**⑦ gridReLoad**

**그리드 재조회는 아래와 같이 요청을 보낼 url을 넣어 코딩**

****

**7.4 View Pattern – 1 Grid + board Write**

****

****

**7.4.1 Board List (Grid) Script**

**① grid setting의 row onSelectRow**

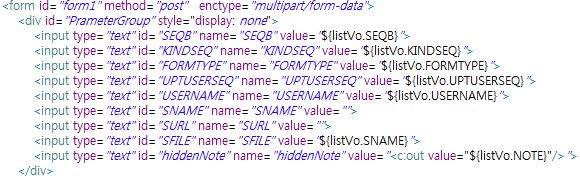
**- 선택한 행의 id(SEQB)를 form에 넣고 해당 폼을 요청할 url로 submit**

****

**7.4.2 Board Write HTML**

**① body**

**- hiddenNote를 추가해 수정된 Note값을 넣어 수정 작업시 사용**

****

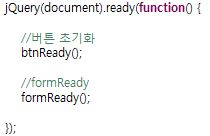
**- 파일첨부 부분 추가**

****

**7.4.3 Board Write Script**

**① 문서를 열자마자 제일 먼저 실행**

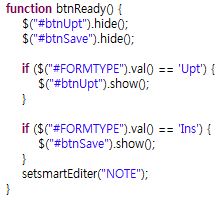
**- 상태에 따라 버튼 visible 여부를 설정하는 btnReady와 form의 초기값을 설정하는 formReady 호출해줌**

****

**② button 초기화**

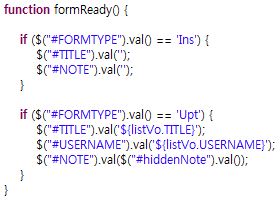
**- 전 화면에서 어떠한 버튼을 클릭하느냐에 따라 FORMTYPE(수정버튼 클릭 시 Upt / 글쓰기 버튼 클릭시 Ins)가 결정**

**넘어온 FORMTYPE(Ins / Upt)에 따라 저장 / 수정 button show / hide 설정**

****

**③ formReady**

**- FORMTYPE(Ins / Upt)에 따라 초기값 설정**

****

**④ button event**

**- 단순 페이지 이동은 $(location).attr(‘href’, ‘url’) 을 이용함**

**- 저장 / 수정에 대한 처리는 FormAction에 정의함**

****

**⑤ 저장 / 수정 처리**

**- 내용 입력 부분은 smartEditor를 이용하며 아래와 같이 코딩**

**- submit은 ajaxsubmit 사용**

****

**⑥ 작업후 페이지 이동**

**- 저장 / 수정에 따라 다른 페이지 이동(Upt 는 게시글 보기 / Ins는 게시글 리스트)**

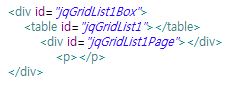
****

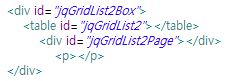
**7.5 View Pattern – 2Grid**

****

**7.5.1 HTML**

**① body에 테이블 2개 생성하여 2개의 그리드를 뿌림**

****

****

**7.5.2 Script**

**① 문서를 열자마자 제일 먼저 실행되는 부분**

**- 입력 / 수정 기능이 있는 폼은 validation check 해줘야 함(formValidate()에 정의함)**

**- jqGrid Setting 부분에 jqGrid 2개 모두 세팅**

****

**② jqGrid 사이즈 조정**

**-그리드 2개 모두 resize**

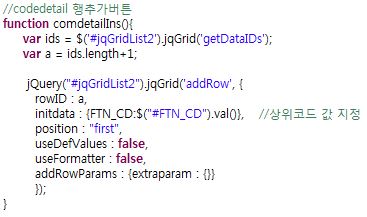
****

**③ row add button event**

**- jqGrid의 addRow 메소드를 이용하여 행을 추가**

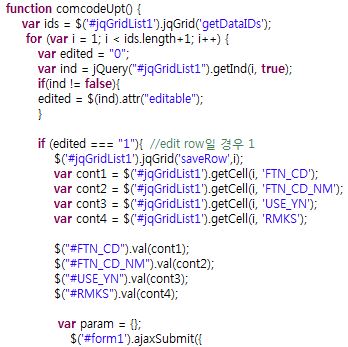
**- addRow의 parameters**

|  |  |
| --- | --- |
| **parameters** | **설명** |
| **rowID** | **새로 추가될 행의 id** |
| **Initdata** | **초기값** |
| **Position** | **새로운 행이 들어갈 위치(first / last)** |
| **useDefValues** | **colModel의 editoptions에 DefaultValue 속성을 사용여부** |
| **useFormatter** | **포맷 작업에서 매개 변수 동기화 여부** |
| **addRowParams** | **addRow에 전달되는 매개 변수** |

****

**④ row save button event / row update button event**

**- 멀티 행 수정 / 입력은 아래와 같이 값을 form에 넣고 ajaxsubmit 을 사용하여 submit함**

****

**⑤ row delete button event**

**- ajaxsubmit 사용하여 처리**

****

**⑥ jqGrid1 Setting**

**- onSelectRow 에 해당 row를 선택하면 jqGrid2가 조회되게 세팅**

**- ondbClickRow 에 해당 row를 더블클릭하면 edit 상태로 변하게 세팅**

**- edit row 설정은 colModel 내 지정하고자 하는 컬럼에 editable : true를 추가해 준다**

****

****

**7.6 View Pattern – Popup**

****

**7.6.1 HTML**

**① Head 부분에 popup css 추가**

**- popup\_background.css파일에는**

**#popup\_background > 배경색(흰색), display : none / #sub > 크기(top250px / left 380px) 가 정의됨**

****

**② Body에 팝업 그리드 추가**

**- 팝업창 내에 그리드 생성**

****

**7.6.2 Script**

**① jqGrid Setting내 double click event**

**- maskfunction호출**

**#popUp\_background를 어떻게 보여줄지 설정**

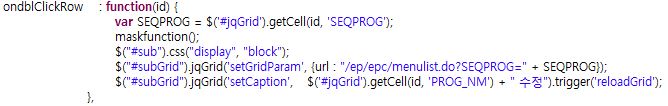
**fadeIn 0.1초의 속도로 천천히 화면에 나타나며 (0.1초 대신 slow / fast 로 대신 입력 가능)**

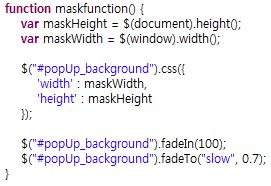
**fadeTo 투명도 0.7 느리게(slow / fast 대신 1000 = 1초로 입력 가능)**

**- 팝업 화면 보여줌**

**display : block은 팝업화면 외 나머지 부분은 보이지 않는 여백으로 채움**

**- 팝업 화면에 기존 정보 받아옴**

****

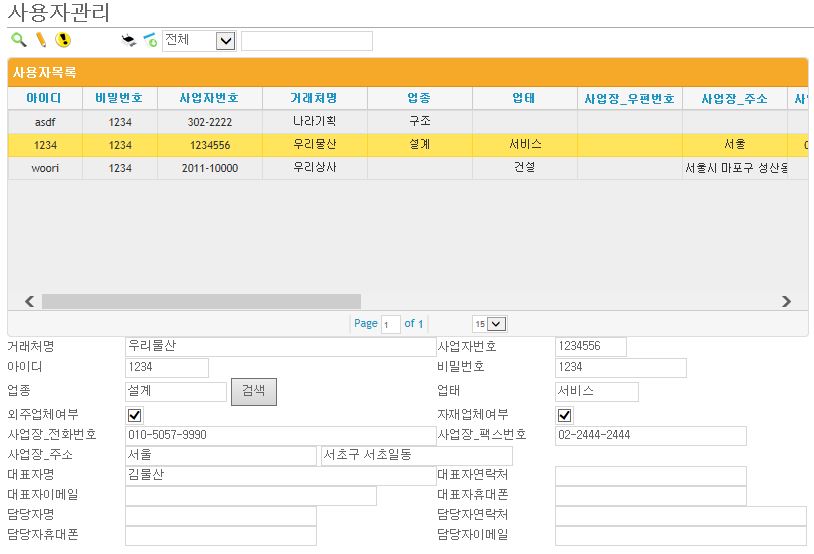
****

**② popup exit**

**- #sub과 #popUp\_background를 보이지 않게하며 reload**

****

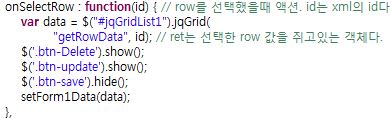
**7.7 View Pattern – 1Grid + 1Form**

****

**7.7.1 Script**

**① grid setting의 row onSelectRow**

**- 선택한 행의 데이터를 setFrom1Data로 넘김**

****

**② setForm1Data(data)**

**- 선택한 행의 데이터를 하단 폼에 뿌려줌**

****