

1. 개요	M찬	7
1.1 목적		7
2. 업무코드정의		8
2.1 AI스현) 코드		8
2.2 단위업무 코드 «...g		
3. 명명표준		10
3.1 기본 명명 STYLE		10
가. Camel Case		10
나. Pascal Case		10
다. Lower Case, •그		10
라. Upper Case		10
마. Hungarian Notation		10
바. Snake Notation		10
3.2 본 프로젝트 프로그램 명명 규칙		11
가. 서버 프로그램 (Java) 명명규칙	그.11	
나. UI J프-로그>권 0A/ebsqudre) 관추		15
3.3 기본 주석 STYLE		17
가. 볼 립 주석		17
나. 단일라인 주석.....요」		17
다 라인 주석 _		18
라. Markup Language 주석		18
3.4 본 프로젝트 주석 명명 규칙		19
가. 서버 프로그램 (java) 주석 명명규칙		19
나. UI 프로그램 (Websquare) 주석 향명 규칙		21
4. 통합개W환경		22
4.1 la		22
4.2 WEBAPPLICATION UBRARIES		22
4.3 ^A/ORKSPACE.....•"... , ", 대...*		24
5. FRAMEWORK		27
6. 코딩 가이드		28
6.1 서버 프로그램 (JAVA) 개발		28
가. Controller		28

다. Service	30
라. SqlMap	32
마. Request & Response 처리	36
바. Transaction	38
시W Stored Procedure	41
아. Exception 처리	42
자. Properties 사용	46
차. Session 처리	47
타. Paging 처리	60
각종 ta 처리	63
6.2.3 프로그램 (WEBSQUARE) 개발	63
가. Frame Set 개요	63
나. as 키	65
다. DataSet 등록 및 편집	66
라. 기본 공통 Script	70
7. TESTCASE	74
7.1 CONTROLLER URL 테스트	74
7.2 SERVICE 합	75
7.3 DATASET (DATACOLLECTION/ DATA서AP)	76
7.5 SAX 테스트	78
7.6 POI 테스트	80
8. 8.1 OZ REPORT	81
가 레포E 생물	81
h Client Report 생성 요청	83
8.2 MARKANY Server Report 생성 요청	85
8.3 JOB-PASS	89
가. 실행	89
다 요	90
라. Job 생성	91
DL Job S3	92

바. OPII-----	93
9. 시큐어 코딩-----	98
9.1 전자정부 CODE INSPCCTION RULESET 39^1-----	98
9.2 일반 코드품질개선 예제-----	99
9.3 코드 가독성 개선 예제-----	102
9.4 코드 설계변경으로 품질개선 예제-----	103
9.5 기타 코드표준 개선 예제-----	108

민간인 /심지함-김두포 /2020-11-07 13:58

1.개요

1.1 목적

본 문서는 국방인사정보체계고도화의 개발표준정의서로서 개발방법 및 개발 표준을 제정하고 가이드를 제시하여 프로그램의 품질 향상과 일관성을 유지한다.

또한 여러 프로그램에서 공통적으로 사용되는 모듈을 컴포넌트화하여, 한번 작성된 컴포넌트의 재사용을 높인다.

이는 개발기간 단축 및 비용절감을 가져오고, 운영자 측면에서 효율적인 시스템 유지보수, 그리고 시스템 사용자측면에서 일관된 구성과 기능을 제공한다.

본 문서는 기록을 목적으로 한 문서가 아니라, 개발자간의 이해와 공유를 위한 문서이므로, 개발자간에 본 문서의 내용을 이해하고, 적용하며 또한 내용 중 변경이 되어야 할 사항이나 추가로 공유되어야 할 사항이 있다면, 피드백을 통해 본 문서에 반영 될 수 있도록 한다.

민자인 /심지형-김두포 /2020-11-07 13:58

2.업무코드정의

2.1 시스템 코드

최상위 시스템코드를 정의,

시스템 명	시스템코드	전체이름
국방인사정보처! 계	DHRMIS	Defence Human Resource Military Information System

민간인 /심지환-김두포 /2020-13:58

2.2 단위업무 코드

단위업무를 Level1,2,3 로 나눠 하위코드를 정의한다.

이 업무코드를 기준으로 Java Package와 Websquare 내 디렉토리 구조를 정의한다.

시스템명	Level1	전체어목	약어	Level2	전 轉 이목	약어	Level3	전轉이 ■	약어
체계통합	SIN	*_greation	r상호운영성	기반轉계	Iteroperdbility	•OP			
				체지면물	Infrastructure	IFR			
				더어타탐석	Dau connectton	DCO			
연애편역	SAdMtpowec	MAN	DW	안소운영계	Data AfMly\$a	DAN			
				1전시언, <<<계획	Data WdrHxxjse	DWH			
				해, 류	Sourcing	SOU			
인사관리	Human	HUM	정관리	전 a 관리	kWortforce Ptar for	NOR			
				기름관리	^ortcfoKe Ptan for	WAE			
				시@/심억관리	Mar	FST			
인사정보체계	A'crk	WAD	지 輕	장기/복무연성	Assign nwfit	ASG			
				지 < 千 천	ApfXIHMl	APR			
				田역관리	httMfIM	PRO			
전시업*	A노rteneWcHt	WAR	聲대병적유지	田역관리	Rwwn	PER			
				전사/순직관리	RMOUFOS Rrwiew	WS			
				전세 역관리	Long-tenn Service	LNG			
鳴轉관리	System	sysa	한관리	의學근學	Comnwid	찌			
				가군 법	^ococnnwndod	DCH			
				전시연사 【1이터 과	Management	AOS			
자량전醫	Mana9«fment	X	직관리	전시연사 【1이터 과	Mfttary Fore着 CM*여	MFD			
				서«스관리	Report	Report			
				자량전醫	MaM09(Twnt	MAI			
자량전醫	System	sysa	한관리	전사/순직관리	MMary R>cc«	MAI			
				전세 역관리	Main ulin	DEA			
				의學근學	OeMh «)Battle	R的			
자량전醫	Mana9«fment	X	직관리	가군 법	너,離양舊T	DUT			
				전시연사 【1이터 과	WMtwrw Military	I AW			
				서«스관리	Register	WO			
자량전醫	System	sysa	한관리	의學근學	Management	AUT			
				가군 법	Duty Work	ORG			
				전시연사 【1이터 과	MMary Oisc^piina	SER			
자량전醫	Mana9«fment	X	직관리	의學근學	Uw	DTR			
				가군 법	flowurces afhirt				
				전시연사 【1이터 과	(XMaFHe				

3. 명명표준

3.1 기본 명명 Style

명명규칙에는 아래와 같이 Pascal/Δel Case, Upper Case, Lower Case 등 여러가지 종류가 있고, 본 프로젝트는 이들을 조합하여 적으로 사용한다.

가. Camel Case

의미있는 단어의 조합에서 각 단어 앞자리는 대문자, 나머지는 소문자로 표기.

첫자리는 소문자로 시작

ex) userList, myName

나. Pascal Case

첫자리를 대문자로 시작하는 Camel Case (aka Upper Camel Case)

ex) UserUst, MyName

다. Lower Case

식별자의 모든 단어를 소문자로 조합한다.

ex) package mil.dhrmissin.iop.web;

라. Upper Case

식별자의 모든 단어를 대문자로 조합한다, 단어들간의 구분자는 underscored 로 구분한다.

ex) MY^NAME

Ph Hungarian Notation

Data 유형을 Prefix 로 붙여놓은 것

ex) strUserName, mapUserList

바. Snake Notation

단어들간의 구분자를 underscorefj)로 구분한다,

ex) my^name

3.2 본 프로젝트S 프로그램 명 명 규칙

모든 이름은 영문, 숫자의 조합 (a-z, A-Z, 0-9)을 원칙으로 한다.

숫자로 시작하지 않는다.

특수기호는 사용하지 않는 것을 원칙으로 한다.

과도한 약어의 사용은 피하고 정확한 의미 전달을 위해 되도록 FullName 을 사용한다.

가. 서버 프로그램 (Java) 명명규칙

(1) Package

- 모든 패키지는 Lower Case Style 로 명명한다.
- 각 Layer 는 아래와 같은 명명규칙으로 한다.
 - mil.dhmis4업무분류 Level1}4업무분류 Level2}. {업무분류 Level3}. web, service, (vo), sqlmap

(2) Class

- Entity + ActionGroup + Layer 명 + java 로 명 명

Entity 명

lyA 업무주제명

- ActionGroup 명

- 해당 Entity * 가지고 처리할 Acton 들을 아우르는 이름

유형	영문명 Full Name	Action Group (약어)
조회	Inquiry	Inqr
입력	Registration	Rgst
처리	Update	Updt
삭제	Delete	Delt
관리	Management	Mgt
처리	Process	Pres
매핑	Mapping	Mpp
배치	Batch	Bat

ex1) select insert, update, delete 등 전체적인 관리를 위한 Action 들의 Group — Manage —◆ Mgt

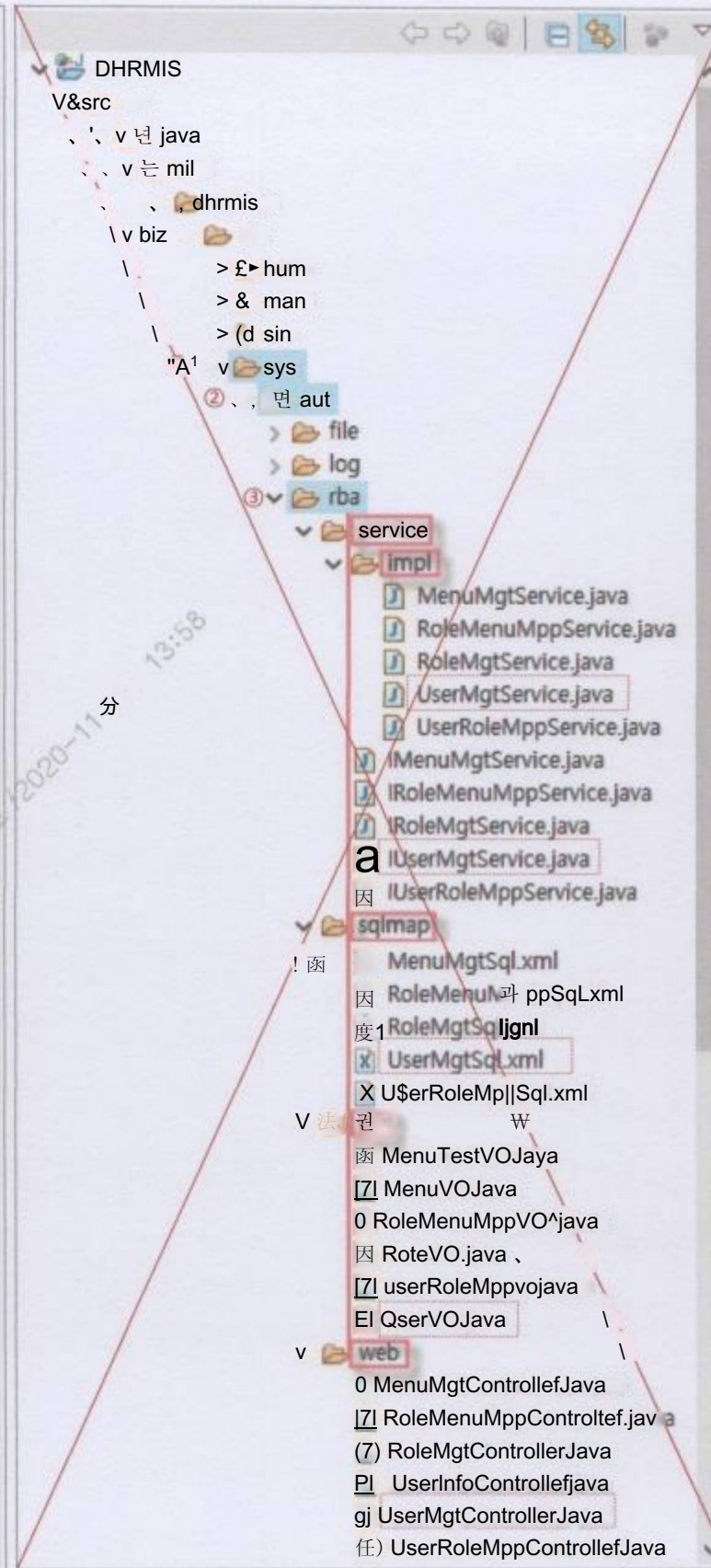
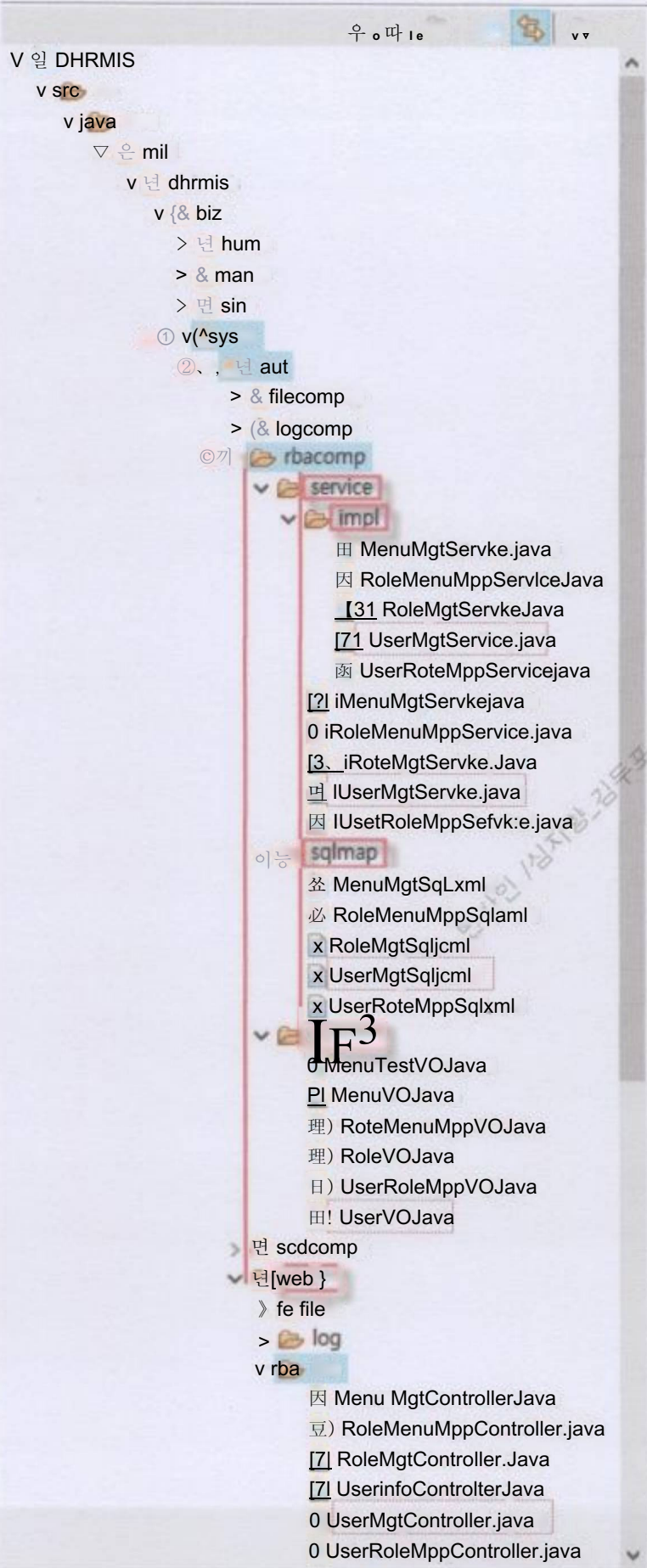
ex2) process 위주의 Action 을이 모인 Action 들의 Group —◆ Process — Pres

- Layer 명

—Controller, -Service, ~Sql

- 예 제

- EmpMgtControllerjava^ EmpMgtService.java, EmpMgtSqlxml
- FileGarbageCollectingPrsController.java. FileGarbageCollectionPrsServke.java, FileGarbageCollectionSqljtml



(3) Method

- 첫문자를 소문자로 시작하고, 두 단어 이상의 조합일 경우 Camel Case 명명규칙을 사용한다.

ex) selectUserUstO

- Method 명 내에는 구분자 《••》 * 등) 를 사용하지 않으며, Method 의 행위를 충분히 설명할 수 있는 의미있는 단어를 사용한다.
- 한번에 여러 개의 목적을 받거나 리턴 하는 경우에는 Method 명의 마지막에 List * 붙인다.

ex) selectUserQ vs selectUserUstO, saveUserO vs saveUserUstO

J%ethod 의 명명은 기본셋을 점해 사용

작업 내용	동사	예제
데이터 조회 (목록)	select	sselectUserUstO
데이터 입력 (목록)	insert	insertUser(1st)
데이터 수정 (목록)	update	updateUser(List)
데이터 삭제 (목록)	delete	deleteUser(List)
입력, 수정, 삭제 동시 처리	save	saveUser(List)
처리	process	processDayClosing
배치	batch	batchDailyTestlob

- Method 명은 일관성있게 명명해야 함. < Intercept나 AOP를 걸 때 메소드명이 일관적이어야 공통관심사항 《AOP》 들을 주입하는데 문제가 없음. 추가적으로 필요한 Method명은 일단 작성해서 Commit해주세요. 검토후 취합하겠습니다.

■ 예제

```
@RequestMapping("/rba/selectUserList.do")
public void selectUserList(MciRequest req, Model model) throws Exception
{
    // get parameter
    Map<String, Object> params = req.getParameters();

    // invoke service method
    List<Map<String, Object>> list = userManagerService.selectUserList(params);

    // response
    model.addAttribute("userList", list);
}
```


(4) Variable

- 첫문자는 소문자로 시작하고, 두 단어 이상의 조합일 경우 **Camel Case** 명명규칙을 따른다.

ex) userList

- 해당 변수의 의미를 명확히 하기 위해 접두어(Prefix)로 그 속성을 명시한다. (Hungarian 표기법)

ex) mapUser, lstUser, objUser 정도

(5) Attribute

- 첫문자는 소문자로 시작하고, 두 단어 이상의 조합일 경우 **Camel Case** 명명규칙을 따른다,

ex) model.addAttribute(*userList^M, lstUseList);

(6) Constant

- 모두 Upper Case 를 사용하고, 단어와 단어사이는 _(Underbar)로 구분한다.

ex) ERROR.MESSAGE

나. 내 프로그램 (Websquare) 명 명규칙

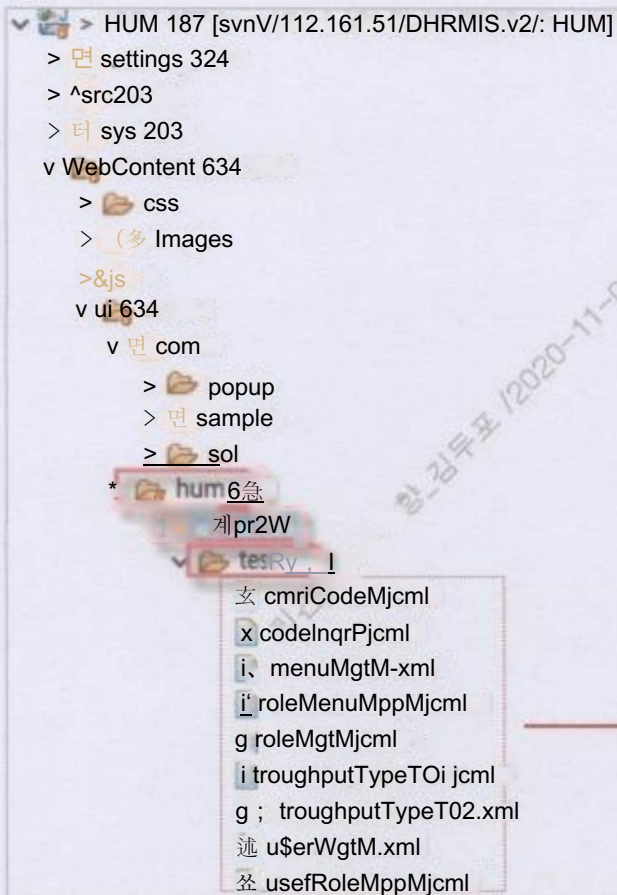
(1) Directory

- 모든 디렉토리는 소문자 《Lower Case Style》로 명명한다.
- 각 디렉토리는 아래와 같은 명명규칙으로 한다.
 - ui / 업무분류Level1 / 업무분류Level2 / 업무분류Level3 / 화면XML

(2) UI

- Entity + ActionGroup + jcml 로 명명

■ 예제



- Mjcml 《메인화면》
- TOUml 《랩》, TOZxml 《랩2》...
- Rxml (팝업)

(3) Variable

- 첫문자는 소문자로 시작하고, 두 단어 이상의 조합일 경우 **Camel Case** 명명규칙을 따른다.

ex) userList

- 해당 변수의 의미풀 명확히 하기 위해 접두어(Prefix) 로 그 여성유 명시한다. 《Hungarian 표기법》

컨트롤타입	접두어	예제
Button	btn	btnSearch
Label	lbl	lblTitle
TextBox	div	divTitle
Combobox (Selectbox)	cmb	cmbMenuLevel
Radio	rdo	rdoSex
InputBox	txt	txtUserName
Text Area	txa	txaContent
Grid	grd	grdUser
TreeView	trv	trvMenu
DataCollection	dl	dlUserList
DataMap	dm	dmParam
Checkbox	chk	chkCode
Inputcalendar	ica	icaDueDay
Chart	cht	chtSale
Generator	gen	genMenu
PageList	pg'	pglBoard
TabControl	tab	tabDetail
WFrame	wfm	wfm Body
Span	spa	spaTitle
Division	div	divFile

(4) Function

- 첫문자는 소문자로 시작하고, 두 단어 이상의 조합일 경우 **Camel Case** 명명규칙을 따른다.

- 변수와 함수를 구분하기 위해 함수 앞에는 **fn** 을 붙인다.

ex) fnSearchUserListO

- Function 영 내에는 구분자(V 등) 를 사용하지 않으며, Function 의 행위를 충분히 설명할 수 있는 의미있는 단어를 사용한다.

단, 이벤트 함수는 V (under bat)를 허용한다.

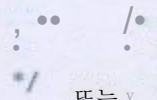
※ 자동으로 생성되는 Event함수들은 자동으로 _(•언더바)가 붙음.

ex) btnSearchj)nclick

3.3 기본 주석 Style

주석은 블록 주석, 단일라인 주석, 라인 주석, Markup주석등 여러 종류가 있고, 본 프로젝트는 이들을 조합하여 복합적으로 사용한다.

가. 블록 주석



또는 (문서주석 vs 구현주석)

블록 주석문은 클래스나 메소드 등을 설명하기 위해 주로 사용한다.

Java API Documentation 0| 넣을 설명은 ex1 과 같이 처리해야 나중에 문서 Generation 시
발체된다, 자체 참조를 위해선 ex2 와 같이 처리

ex1)

```
/**
 * XX 데이터 저장결과 조회
 *
 * @author dp.kim
 * @since 2020.05.29
 * @param MciRequest 저장할 데이터
 * @param Model 저장 결과
 * @return
 * @throws Exception If an exception occurred
 */
```

ex2)

```
/**
 * TODO
 */
```

나. 단일라인 주석

.. */

한줄 주석.

Indent 는 해당코드와 같은 Depth 로 맞춘다.

ex)

```
if (condition)
{
    // Handle the condition. */
    inserttestO;
}
```


다. 라인 주석

//

한줄 주석.

라인 전체 또는 라인의 일부분을 주석처리 할 때 사용.

ex)

```
if (foo > 1)
(
    // Do a double-flip.
    ***
    |
    Else
    <
        return false; // Explain why here.
    )
```

라. Markup Language 주석

<!-- -->

HTML 및 XML 에서 사용하는 주석으로, Markup 언어 내에서 사용할 수 없는 꺾쇠문자 (<>) 는 <, > 쿼 사용한다.

Indent 는 해당코드와 같은 Depth 로 맞춘다.

ex)

```
<!--사용자 정보 조회-->
<select id="selectUserList" parameterType="map">
    /* 사용자 정보 조회 */
    ***
</select>
```

3.4 본 프로젝트 주석 명명 규칙

모든 소스는 **Java API Documentation** 생성에 부합하도록 주석운을 작성한다.

(참고: <http://www.oracle.com/technetwork/java/codeconvtoc-136057.html> > 5 - Comments)

가. 서버 프로그램 (java) 주석 명명규칙

(1) Class

- Class 선언문 전에 위치하고 주석 형식은 문서형 블럭주석(/**... */)을 사용한다.
- 클래스 설명은 가능하면 5, 6 줄 정도로 자세히 기술한다.
- 클래스에 대한 설명, 작성자명, 생성일자 등을 기술한다.
- 예제

```
/**
 * 해당 시스템에 접속하는 전체 사용자를 관리하는 컨트롤러로
 * 본 컨트롤러는 요청 URL과 Sendet을 Mapping해주고, 그 처리결과를 응답해주는
 * Class 이다.
 *
 * @author dp.kim
 * @since 2020.05.29
 */
@Controller
public class UserMgtController
```

(2) Method

- Method 선언문 전에 위치하고 주석 형식은 문서형 블럭주석을 사용한다, (/** ... */)
- 메소드 기능 설명은 한두 ff로 간결하게 기술한다.
- 메소드에 대한 설명, Parameter, Return Type, 작성자명과 생성일자, 예외 등을 기술한다.
- 값이 없는 경우 빈값으로 두지 말고 아예 삭제한다.
- 예제

```
/**
 * 시스템에 접속가능한 사용자 목록을 조회.
 *
 * @author dp.kim
 * @since 2020.05.29
 *
 * @param MciRequest 조회 조건
 * @param Model 조회 결과
 * @throws Exception 병cs:
 * 용throws Exception If an exception occurred
 */
@RequestMapping("/rba/selectUserList.do")
public void selectUserList(MciRequest req, Model model) throws Exception
```


(3) SQL

<1- 사용자정보 조회 ->

<select id«"setectt/serlist" parameterType*"map" resultType«"egovMap">

I com. rbacomp.UserMgt. seLectUserList ' / WAS에서 Log를 추적할수 있게 하7

위해 추가

select T1.emplyr_id,

T1.orgnzt_id

...

from COMTNEMPLYRINFO T1

where 1=1

</select>

Z

민간인 /심지환-김두포 -07 13:58

나. UI 프로그램 (Websquare) 주석 명명 규칙

(1)내

화면에 대한 메뉴경로, 화면명, 설명, 작성자명, 생성일자 등을 기술한다.

```

/*
-----
며 뉴si      :      공릉-사율자-사플땡3라
화면어*     :      사& 자관리
쫓면뵤명    :      사률자 관리
오성자      :      dp.kin
작성일      :      2018.07.24
-----
*/

```

(2) Function

Function 에 대한 설명, Parameter, Return Type. 작성자명과 생성일자, 예제 등을 기술한다.

```

/**
◆ main.xml의 컨텐츠영역의 화면을 이동시킨다.
◆
• 용since 2026.06.10
• ^author 전기완
事 숨param {String} psTitle 화면타이틀
• 을param {String} psSrc 화면 src
• ^param {JSON} poParam 화면에 넘길 그SON과라•에터
• ^return {void}
• ^example
• 메뉴가 목록화면일때 등록화면으로 이동시
• com.changehtenu(•프로그래동좌"ui/sample/programRegister.xml",
{"°AAAH:Maaa% MBBB:Mbbb"});
*
_ 등록화면에서 목록화면으로 이동시
* com.changeMenuCnull^ mill; {"•DDD":Mdddw, "CCC":Mccc"});
*/
com.changeMenu ■ function(psTitle, psSrc, poParam)

```


4. 통합개발환경

4.1 개요

DHRMISIDE.7Z 으로 제공되고, 압축을 풀면 해당 디렉토리 안에 개발과 관련된 Jdk , Eclipse, Tomcat
Ant, Maven, Repository, Workspace 등이 모여있음.

v' 로g 디스크 (C:)

v DHRMISIDE

발관련 ToolSet 통합 eclipsejee-2019-06-R-win32-x86_64
> H)dk
> jobpass
> repository
> server
> tmp
> tools
Workspace > workspace

토털 디스크 (C:)

DHRMISIDE

Java , UI > 3 eclipse-jee-2019-06-R-win32-x86.64
v jdk
> jdk1.8.0.231
> openjdk-1 <wirxiows-x64_b4n
Schedule 齋 j jobpass
> 소 repository
v server
少 어 apdche-tomcat-8.0.36
> mariadb-103.11-Win32
> tmp
, tools
> apache-ant-1.10.4
> apache-cxf-3.3.4
> apdche-maven-3.5.4
> sqllite
v ; workspace
> , metadata
Project > DHRMis

M Eclipse 2019.06 (Eclipse v4.12) - Eclipse Photon (Eclipse v4.8) 이후 연월로 명명

Open JDK 1.8 (Redhat), Oracle JDK 1.8

Subversion- Polarion Suversive Team Provider

vs Tigris Subclipse

OZ는 개설해서 개발

4.2 WebApplication Libraries

수.輪 1.0 今 ト ▽

V 變 WEB-INF

- > 變 dases
- > 變 config
- > 은 files
- > 白 切>
- v 變 lib
 - egovfrfn«wof1cfte.batcofe-3.9.0.jar
 - egovfranwwodcrte.fdl.access-3.9.0jar
 - egovframewodcrte.fdl.cmmn-3.9.0.jdr
 - egovframewodc_rtefdl.cfypto-39.0Jar
 - egcMramewock-rte fdl.excel-3.9.0 jar
 - egovfra(new(x1(.rte.f(a.fitehandling-3.9.0.jar
 - egovfranewodcrte.fdl idgnr-3.9.0 jar
 - egovfraneworLrte.fdl.logging*3.9.0.jdr
 - egovframewock.rte fdl property-3.9.0Jar
 - egovframewock.rte.fdl.securtty-3.9.0.Jdr
 - egovframework.rte.fdl.string-3.9.0Jar
 - egovfratnewodcfte.fdljcrnl-3.9.0Jar
 - e9Ovframewodcrte.itiIntegration-3.9.0jar
 - Q mybatis-3.4.6.jar
 - Q mybatis-spring-1.3.2Jar
 - Q ojdbc8.jar
 - D poi , 3J0 , , ANAL.Jdf
 - Q poi-contrib-3.7-beU3.Jar
 - Q poi-ooxml-3.9.Jar
 - Q poi-ooxml-schemas-3.9jaf
 - Q quartz-2.22.jar
 - Q quactz*jobs-2.22.jar
 - B 씨 4 apl.1.7.7>
 - n spring.aop-4.3.22.RELEASEjar
 - Q spring-aspects-4.3.22.RELEASEjar
 - Q spring-batch-«xe-3.0.6.RELEASEjar
 - Q \$pdng-batch-infra\$tructure-3.0.6.RELEASEJar
 - Q spring-batch-integfation-3.0.6.RELEASEJar
 - Q spring-beans-4.3.22RELEASEJar
 - Q spring-<ontext-4.3.22.RELEASEJar
 - Q spring-context-support-4.3.22.RELEASEjar
 - Q spnng-core-4.3.22.RELEASEjar
 - Q spring-expression-4.3.22RELEASEJaf
 - Q spring-instrnjment-4.3.22.RELEASE.jar
 - Q spring-securtty<onflg-4.2.11 RELEASEJar
 - Q spring-security-core-4.2.11 RELEASEjar
 - Q spnng-security-web-4.2.11 RELEASEjar
 - Q spdng-tx-4.3.22.RELEASE.jar
 - spdng-web-4.3.22.RELEASEJar
 - spring-webmvc-4.3.22.RELEASEjar
 - spring-webmvc-portiet-4.3.22.RELEASEjdr
 - Q spring-websocket-4.3.22.RELEASEjar
 - websquare.5.0J.3487B.20190213.175059.15.j. 七 내
 - Q we^3dapter.5.0J.223IB20160722.155h

→ RFP에 최신 전자정부프레임워크 - 3.9

→ Mybatis 3.4.6

→ Excel Library

→ 스케줄러 (JobPass 지원)

→ Logging

→ 전자정부프레임워크 3.9 Dependency의 SpringFramework 4322

→ SpringSecurity 4.2.11

websquare → adapter

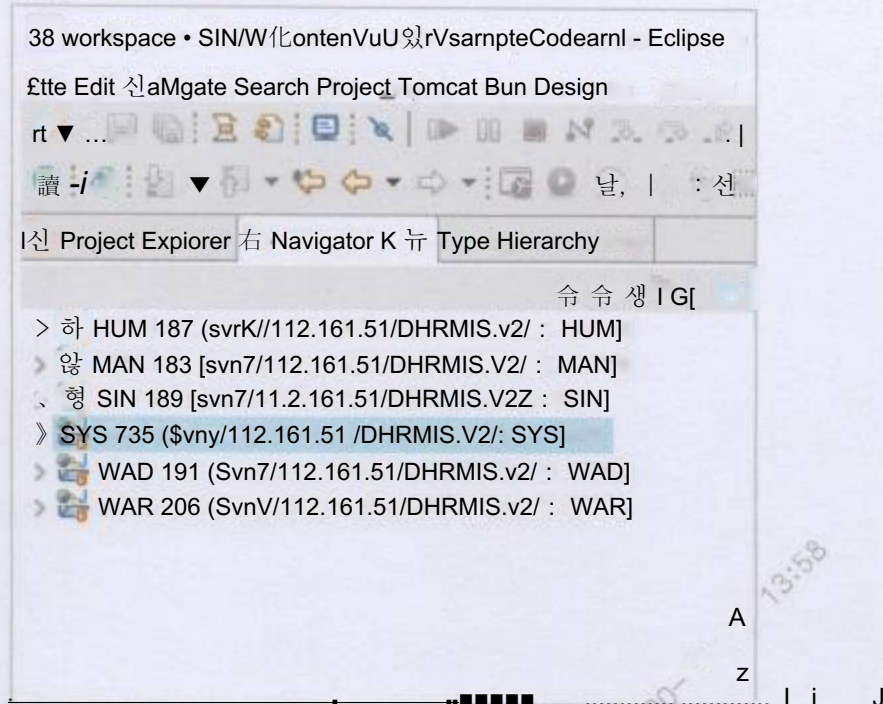
교배 M유 AJ

A름년느,

23/10

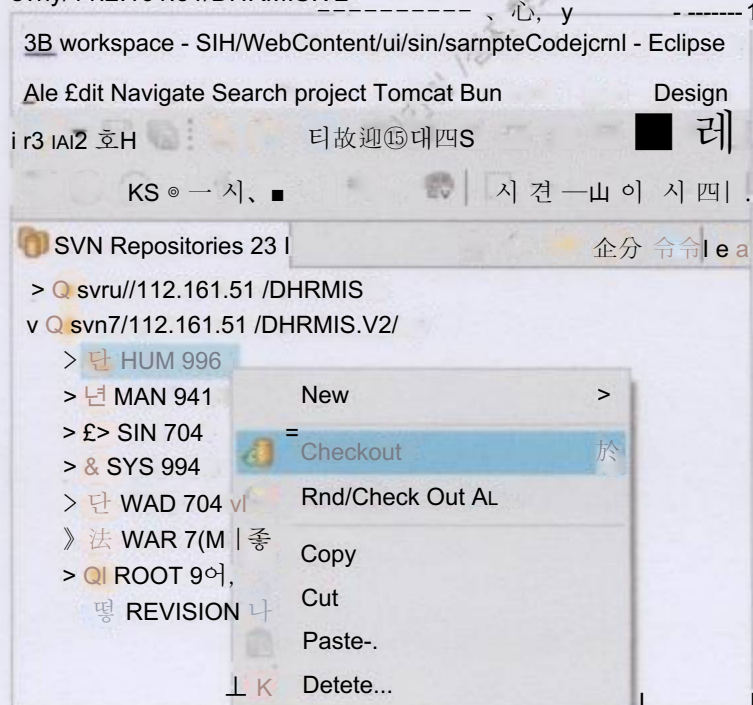
4.3 Workspace

1. 파트별 프로젝트가 분리



2. 아래 주소에서 해당 프로젝트와 SYS 프로젝트 Checkout

svnny/11.2.161.51/DHRMIS.V2



3. 업무별 java 소스 작성 폴더오hui 작성폴더 확인

V 일 HUM 187 [svny/112.161.51 /DHRMIS.v2/ ; HUM]

、 단 settings 324

v src 203

▼ 味 java 2的

、 , (러 mil 203

▽ 평 dhrmts 203

v biz 203

、 , 하 ham 203

> 203

203

203

203

dc203

lng 203

pw 203

203

rec

res

resources 700

> test 203

> sys 203

> fit tmp 704

v 베 WebContent 634

> & css

> images

> 년 js

▼ 味 ui 6S4

> 면 com

v 와 bum 634

> apt 209

> e»g634

□좌 dch 203

와 lng 203

per 203

法상 pro 203

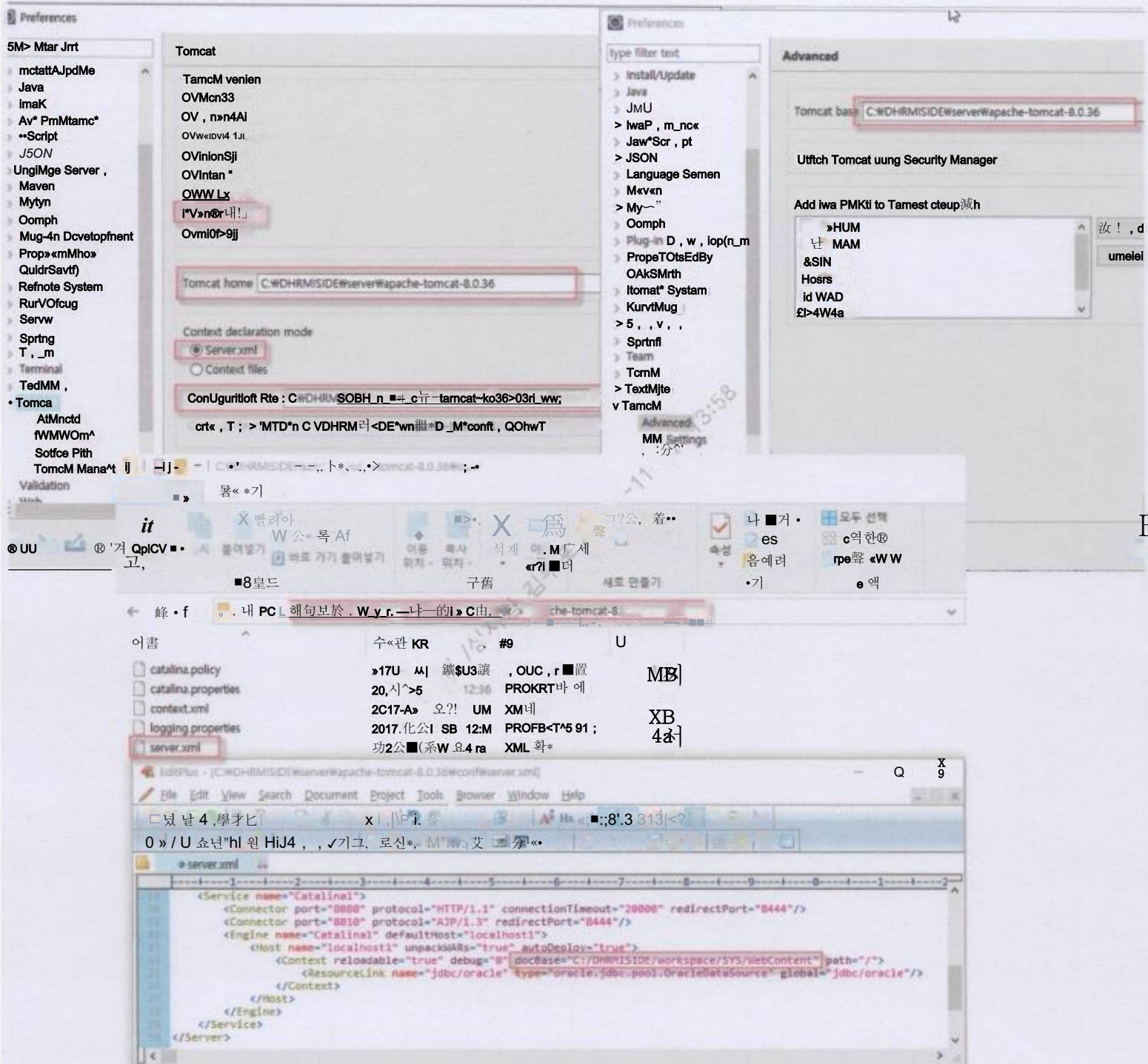
g을 rec 203

res 203

민간인 /심지환-김두포 /2020-11-07 13:58

5. 서버구동 설정

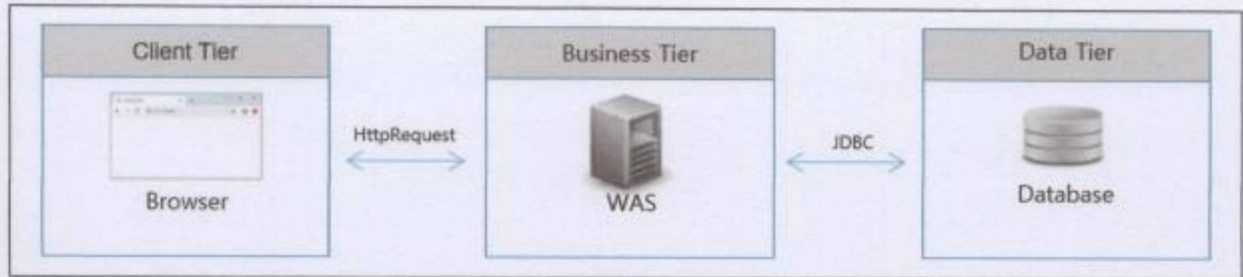
- Eclipse 뿔켓 Plugin 설정
- Tomcat 서버 설정



X 서버 구동시 자신 프로젝트만 띄우고자 하면 dispatcher-servlet.xml, context-common.xml scan범위 수정

5. Framework

그림 1



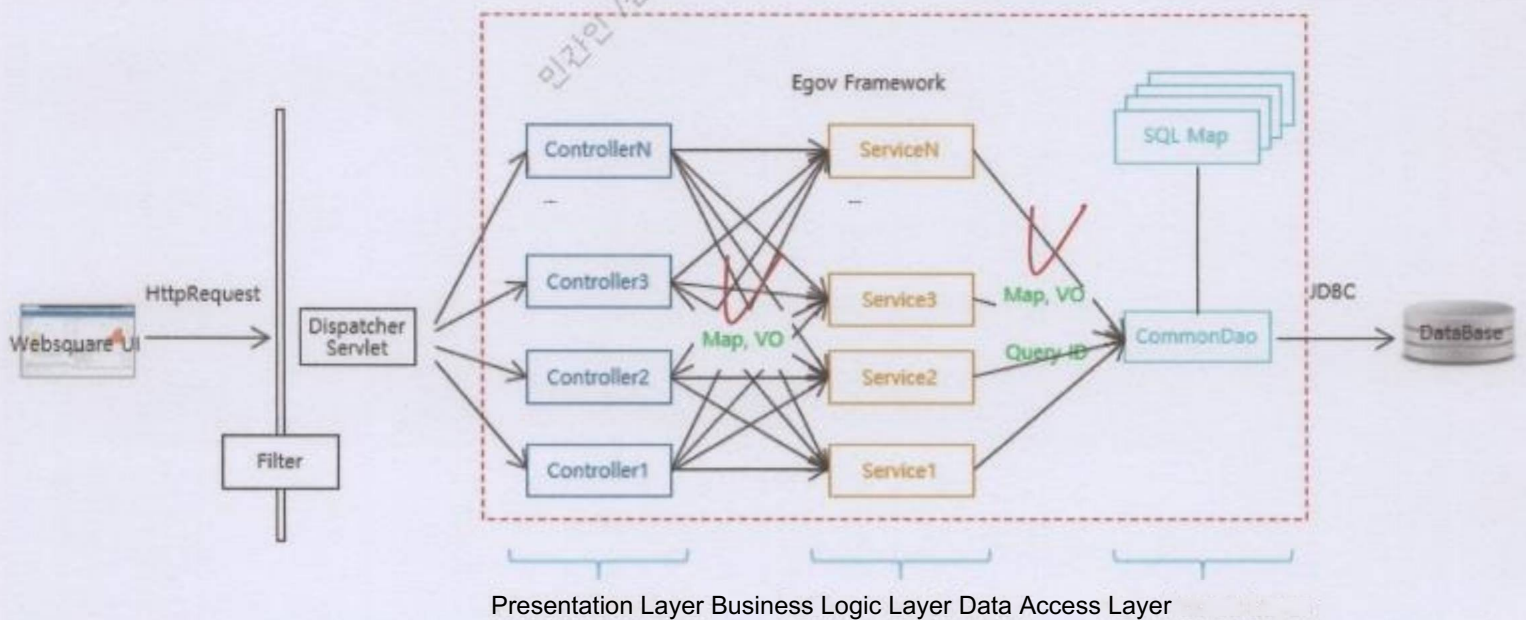
<< System View >>

그림 2



<< Software View >>

그림 3



<< Framework >>

6.코딩 가이드

6.1 서버 프로그램 (Java) 개발

- 해당 프로젝트에서는 Controller, Service, SqlMap 만 작성하고 Dao 는 CommonDao 를 사용한다.

가. Controller

(1) 개요

Controller는 화면처리를 담당하는 구현체로, 단순히 내로부터 넘겨받은 Parameter로 Service를 호출하고, 결과를 다시 화면에 전송하는 역할을 한다.

(2) 작성요령

- 개발자가 작성해야 할 Package, Class 명은

mil.dhrmis ◆ 《업무분류 Level1M 업무분류 Level2}.web.{업무분류 Level3} .{Enttty + ActionGroupJ.Controller.java

- 등록을 위해 별도의 환경설정파일을 필요로 하지 않는다.

- Spring Framework 2| Bean 으로 등록을 위해 @Controller 라는 어노테이션을 Implementation Class 앞에 ■여준다.

- @RequestMapping 을 이용해 호출경로를 지정한다. Class 에 붙어있는 @RequestMapping 경로와 Method 에 붙어있는 @RequestMapping 경로가 조합되어 URL 이 만들어 진다.

- 특별한 예외사항이 없는 경우 Controller 의 파라미터는 MciRequest 와 Model 두개의 파라미터로 처리한다.

```
ex) public void selectUserlist(MciRequest req, Model model)
```

X MciRequest는 Grid Data Request를 받기 위한 Websquare UI Adapter# Wrapping한 Class로

표준S 파라미터를 받는 getParameter(String paramName) , getParameterValues와 같은 메소드

나 威을 기본으로 제공하면서, 추가로

getGridMap(String datasetName)

getGridVO(String datasetName)

등과 같은 그리드 메소드를 제공한다.

X Model은 Service로부터 받은 결과데이터를 내에 내려줄 때 사용된다.

- 예외처리는 무조건 메소드 밖으로 Throw 시킨다. 프레임워크에서 처리.

오류 메시지의 변경을 위해서 try ~ catch 구문 사용은 허용하나 여전히 NestedExcetion 은 밖으로 Throw 시킨다.

K 유의사항

Controller에서 Service에 처리물 위임할 때 Service Method는 반드시 하나만 호출한다.

만약, 두개 이상의 Service Method를 호«하고 싶을 경우 두개의 Service Method! 하나로 묶는 별도의 Service Method 만꺽어 호출한다.

: 3) 예 제

```

~7^
• User<gt;Controller Provides infr>structure for determining view nches fro< URLs
•
• <P>
• 자세한 설@
• *author dp.kl*
• *Sficc 292e.es.29
•
-Controller
public class UserMgtController
{
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    private UserMgtService userMgtService;

    /**
     * Returns User Information. Request must specified
     *
     * <P>
     * 시스템 접속가능한 사용자 목록 조회
     * *author dp.kl*
     * *Unci 2020.85.29
     * *param MclRequest 조회 조건
     * *param Model 저장 정보
     * *return
     * *throws Exception If an exception occurred
     */
    @RequestMapping("/rba/selectUserList.do")
    public void selectUserList(MclRequest req, Model model) throws Exception
    {
        // get parameter
        Map<String, Object> params = req.getParameterMap();

        // logger.debug("**.. * * paraas.t*>String());
        // irwpke 의엔. *et-0
        List<Map<String, Object>> list = userMgtService.selectUserList(params);

        // response
        model.addAttribute("userlist", list);
    }

    /**
     * Returnl Result Information having. Request must specified
     *
     * <P>
     * 시스템 접속가능한 사용자 목록을 저장
     * *author dp.kl*
     * *Unci 2020.85.29
     * *param MclRequest 저장할 데이터
     * *param Model 저장 정보
     * *return
     * *throws Exception If an exception occurred
     */
    @RequestMapping("/rba/saveUserList.do")
    public void saveUserList(MclRequest req, Model model) throws Exception
    {
        // get parameter
        List<Map<String, Object>> userList = (List<Map<String, Object>>) req.getParameterMap("userList");

        // invoke service method
        userMgtService.saveUserList(userList);
    }
}

```

17-----

X Autowired vs Resource

- Autowired 는 Spring 전용 Annotation 이고, Resource 는 JSR-250 에 따른 JDK Standard Annotation.

• Resource 를 사용하려면 《name=_userMgtService*》와 같은 속성을 더 지정해줘야 하는 조금 더 귀찮음이 존재. □Type 으로 찾냐 Name 으로 찾냐의 차이)

나. Service

(1) 개요

- **E**ntityClass는 업무로직이 쓰여지는 구현체로, Controller로부터 넘겨받은 Parameter# Mybatis, Id와 함께 CommonDao를 호출하고, 결과를 다시 Controller에게 반환한다.
- Action의 단위가 된다.

(2) 작성요령

- 개발자가 작성해야 할 Package, Class 는 Interface 와 Implementation 2 개.로 아래와 같은 명명규칙을 갖는다.

mil.dhmi 니업무분류 Le선1M 업무분류 Level2}. {업무분류 Le 化 13+comphserviceJ + {Entity + ActionGroupJServicejav
mil.dhmi 이업무분류 Level1}, {업무분류 Levd2}. 《업무분류 Levd3+comphsefAeimpl 仏 + ActionGroup}5servicejavc

- 등록을 위해 별도의 환경설정파일을 필요로 하지 않는다.

- Spring Framework 의 Bean 으로 등록을 위해 @Service 라는 어노테이션을 Implementation Class 앞에 붙여준다.

- 전자정부 표준 프레임워크와의 호환성을 위해 EgovAbstractServiceImpl 을 extends 한다.

- 예외처리는 무조건 매소드 밖으로 Throw 시킨다. 프레임워크에서 처리.

오류 메시지의 변경을 위해서 try - catch 구문 사용은 허용하나 여전히 NestedExcetion 은 밖으로 Throw 시킨다.

(3》에 제

1. Interface

```
public interface IUserMgtService
{
    public Map<String, Object> loadUserByUserNm(String id, String pw);

    public void saveUserList(List<Object> lstData);
}
```

2. Implementation

```
/**
 * UserMgtService is a concrete implementation of the AbstractEgovAbstractService interface class
 *
 * <P>
 * 자세한 설명
 * IM*thor dp.kia
 * 을 &inc 2626.05.29
 */
@Service
public class UserMgtServiceImpl extends EgovAbstractServiceImpl implements IUserMgtService
{
    *Auto_drod
    private CommonDao dao;

    /**
     * Returns User Information. Request must specified
     *
     * <P>
     * 자세한 설명
     * IM*thor dp.kia
     * @since 2626.05.29
     * @param String 조회 조건
     * return Map 조회 결과
     */
    public Map<String, Object> loadUserByUserNm(String id)
    {
        return dao.queryForObject(<?LMet*MgtSql.포selectUserfyld*, {Mran);
    }

    /**
     * Returns Result Information for Saving. Request must be specified
     *
     * <p>
     * 자세한 설명
     * IM*thor dp.kia
     * @since 2020.29 05.
     * 하ar_List 저장tt Q어타
     * 參r, tv轉
     */
    *Transaction사(propagation = Propagation.*EQI/IXED)
    public void saveUserList(List<Object> lstData)
    {
        for(int i = 0; i < lstData.size(); i++)
        {
            Map<String, Object> userData = (Map<String, Object>) lstData.get(i);
            String rowSt = userData.getRowStatus();

            if(rowSt.equals(RowStatus.INSERT))
            {
                dao.insert(<?User*MgtSql.insertUser*, userData);
            }
            *1st if(rowStatus.equals(RowStatus.UPDATE))
            {
                <?Uo.update(*U4erMgtSql, updateUser*, userData);
            }
            *! if(rowStatus.equals(RowStatus.DELETE))
            {
                dao.deleteC-UserMgtSql.deleteUser*, userData);
            }
        }
    }
}
```

다, CommonDAO

《1》 개요

전자정부 표준프레임워크와의 호환성을 위해 EgovAbstractMapper를 extends 하였고,
단순 DataBase 조회, 입력, 수정, 삭제 이외에 File Upload, Download등의 부가기능을 제공한다.

(2) API

CommonDao의 Method는 기본적인 DataBase CRUD와 관계된

queryForList

queryForObject

queryForString

queryForInteger

insert

update

delete

batchinsert

batchUpdate 가 있고,

Query수행후 File을 Octet-stream으로 바로 다운로드 받을수 있게

downloadCsvFile, downloadXlsFile, downloadXlsxFile 메소드 외

upload와 f 려한 메츠드도 js드xls, xlsxg려」가크 에은트를 제공으다. 아래 스나. File처리'에서 설 명

CommonDao APIs	설 명
List queryForUstfString statementName)	statecnentName으로 니st형태의 다건 데이터읽 조회한다.
List queryForUstfString statementName, Object parameterobject)	stateinentName과 parameterObjectS List형태의 다건 데이터읽 조회한다,
Object queryForObject(String statementName) -----겨*, -----	statementName으S 0어«1형태의 단건 데이터■ 조회한다.
Object queryForObject(String statementName, Object parameterobject)	statementName과 parameterObject* 0어ect형태의 단건 데이터읽 조회한다.
int insert(String statementName)	statementName으S 데이터읽 입력한다.
int insert(Stfing statefnentName, Object parameterobject)	statementName과 parameterObjectS. 데이 터를 입력한다. insert 후에 paramObject 로부터 다시 getter* 통해 PK> 받을 수 있다.
int update(String statementName)	statementName으로 데이터읽 수정한다. 리S되는 값은 수정결과 변경된 Row Count가 넘어온다.
int update(String statementName, Object parameterObject)	statementNam공과 parameterobjects. 데이터■ 수정한다. 리턴되는 값은 수정결과 변경된 Row Count가 넘어온다.
int ddete(String statementName)	statementName으로 데이터읽 삭제한다. 리턴되는 값은 삭제결과 변경된 Row Count가 넘어온다.
int deletefString statementName, Object parameterObject)	statementName과 parameterObjectfi 데이터■ 삭제한다. 리턴되는 값은 삭제결과 변경된 Row Count가 넘어온다.
void downloadCsvFile(String statementName, Map param, String fileName, HttpServletResponse res)	statementName-2 param으1 조회한 다건의 데이터■ Csv파일로 스트리밍 다운로드한다.
void downloadXlsFilefString statementName, Map param String fileName, HttpServletResponse res)	statementName과 param으로 조회한 다건의 데이터■ XE파일로 스트리밍 다운로드한다.
void downloadXlsxFilefString statemeotName, Map param, String fileName, HttpServietResponse res)	statementName과 param으로 조회한 다건의 데이터■ Xlsx파일로 스트리밍 다운로드한다.

라. SqlMap

(1) 개요

본 시스템은 ORM (Object Relational Mapping)으로 MyBatis를 선정했다.

Query를 Java소스로부터 분리함에 따라 업무 로직에 집중하게 함으로써 개발자의 생산성 향상 및 업무 효율을 높여준다.

(2) 작성요령

■ 개발자가 작성해야 할 Sqlmap 및 경로는

mil.dhrmis.{시^110}.{Level1}.{Level2}.{level3}comp.sqlmap.{Entity + ActionGroup}Sqlxml 이고 ! , 별도의 환경설정파일 등록은 필요하지 않다.

■ 동적쿼리 작성시 적절한 Tag 와 Attribute 를 사용한다.

■ 조회쿼리 작성시 컬S 개수를 동적으로 분리하는 분기처리는 절대 금지.

(최초 1회 실행시 만들어진 ResultSet 컬럼개수로 컴파일 되어, 이후 더 많은 컬럼이 요청될 때 오류발생)

ex) select

~~<if tets="xxx">emp_no</if>
<if tets="xxx">emp_name</if>
<if tets="xxx">dept_id</if>
from EMP~~

■ Tag

유형	성명
<if>	조건
<choose> <when> <otherwise>	선래
<trim>	Where 8에서 먼 앞에 and나 or 없애주는 태그
<foreach>	루프

■ Attribute

속성	설명
test	조건비교
prefix. prefixOwerides	Where 설 앞에 and나 oHI 처리

■ Parameter 와 ResultSet 에 대해 MyBatis 가 기본적으로 제공하는 Alias 와 사용자정의 Alias 를 나열하였다. (ibatis TypeAliasRegistry 및 sqlmap-config.xml 에서 제공)

alias	type
string	java.lang.String
byte	java.lang.Byte

long	java.lang.Long
short	java.lang.Short
int integer	java.lang.Integer
double	java.lang.Double
float	java.lang.Float
boolean	java.lang.Boolean
date	java.util.Date
decimal	java.math.BigDecimal
object	java.lang.Object
map	java.util.Map
hashmap	java.util.HashMap
list	java.util.List
arraylist	java.util.ArrayList
collection	java.util.Collection
iterator	java.util.Iterator
cameCaseMap	egovFramework.rie.psi.aiaacce^s.uni.tgovhnap
ordered Map	java.util.LinkedHashMap

마. Request & Response 처리

(1) Request 처리

- Controller 에서 특별한 경우를 제외하곤, Parameter 로 HttpServletRequest 을 쓰지 말고 MCIRequest 사용
- 본 시스템에서 제공하는 MCIRequest 는 사용자의 요청이 Websquare 이든, Nexacro 든 기타 상용 UI Framework 이든 jsp 이든 표준웹이든 다양한 UI 클라이언트와 연동할 수 있도록 하는 멀티채널서비스 이다.

Method	설 명
String getParameter(String paramName)	지정한 파라미터명으로 String 형태의 데이터를 받는다.
Map getParam(String datasetName)	지정한 데이터셋명으로 Map 형태의 데이터를 받는다.
Map getMap(String datasetName)	지정한 데이터셋명으로 Map 형태의 데이터를 받는다.
list getGndMap(String datasetName)	지정한 데이터셋명으로 List<Map> 형태의 그리드데이터를 받는다.
List getGridVO(String datasetName, Class ds)	지정한 데이터셋명으로 List<VO> 형태의 그리드데이터를 받는다.
void fileUploadO	파일 업로드
List<FileVO> getFileListO	업로드한 파일정보 목록 취득

민간 - /심지함-김두포 /2020-11-11 14:50

、 於

(2) Response 처리

■ 표준 웹 데이터 통신시 《text/html 로 응답》

```

@Controller
public class UserMgmtController{

    @RequestMapping("/rba/userMgt")
    public String selectUserList(MciRequest req, Model model) throws Exception
    {
        // get parameter
        Map<String, Object> param = req.getParam("param");

        // invoke service method
        List<Map<String, Object>> list = userMgmtService.selectUserList(param);

        // response
        model.addAttribute("dsUserList", list);

        return "/userMgt";
    }

    @RequestMapping("/rba/userMgt2")
    public ModelAndView selectUserList(MciRequest req, Model model) throws Exception
    {
        // get parameter
        Map<String, Object> param = req.getParam("dsParam");

        // invoke service method
        List<Map<String, Object>> list = userMgmtService.selectUserList(param);

        // response
        ModelAndView mv = new ModelAndView();
        mv.addAttribute("userList", list);
        mv.setView("/userMgt");

        return mv;
    }
}

```

→ 지정한 String의 문자열 이름의 jsp페이지로 Forwarding.

→ 지정한 View 이름의 jsp로 Forwarding하고, userList라는 속성명으로 데이터를 Return.

■ Ajax 데이터 통신시 (json data 로 응답)

```

package mil.dhmis.oes.biz.ct.cncl.cmgt.web;

@Controller
public class UserMgmtController
{
    @RequestMapping("/rba/selectUserList")
    public void selectUserList(MciRequest req, Model model) throws Exception{
        // get parameter
        Map<String, Object> param = req.getParam("dsParam");

        // invoke service method
        List<Map<String, Object>> list = userMgmtService.selectUserList(param);

        // response
        model.addAttribute("userList", list);
    }
}

```

→ 리턴페이지가 없는 경우 void로 처리한다.

마. Transaction

(1) 개요

Bisuness 처리중 오류 발생시 전체 Commit 또는 일부 Commit 또는 전체 Roj/back등의 처리.

(2) 작성요령

■ 트랜잭션 처리를 위한 여러방법이 있는데 (Programming 방식 vs Declarative 방식)

본 프로젝트는Annotaion 선언방식을 사용한다.

트랜잭션처리 f f 해 @Transactional 이라는 어노테이션을 Service Method 선언부 상단에 적어주는데 경우의 수는 아래와 같이 여러가지가 있다.

Transactional(propagation=Propagation.REQUIRED),

Transactional(propagation=Propagation.REQUIRES.NEW)

Transactional(readOnly=true)

Transactional(propagation=MANDATORY),

Transactional(propagation=SUPPORTS),

Transactional(propagation=NOT_SUPPORTSI

Transactional(propagation=NEVER),

Transactional(propagation=N ESTED)

■ Propagation은 트랜잭션 전파에 관한 설정으로

하나의 서비스가 다른 서비스를 호출하고 그 서비스가 또 다른 서비스를 호출하는 등 여러 서비스가 하나로 묶였을 때 해당 트랜잭션의 전파에 관한 설정.

o반적으로 트랜잭션이 필요할 경우에는

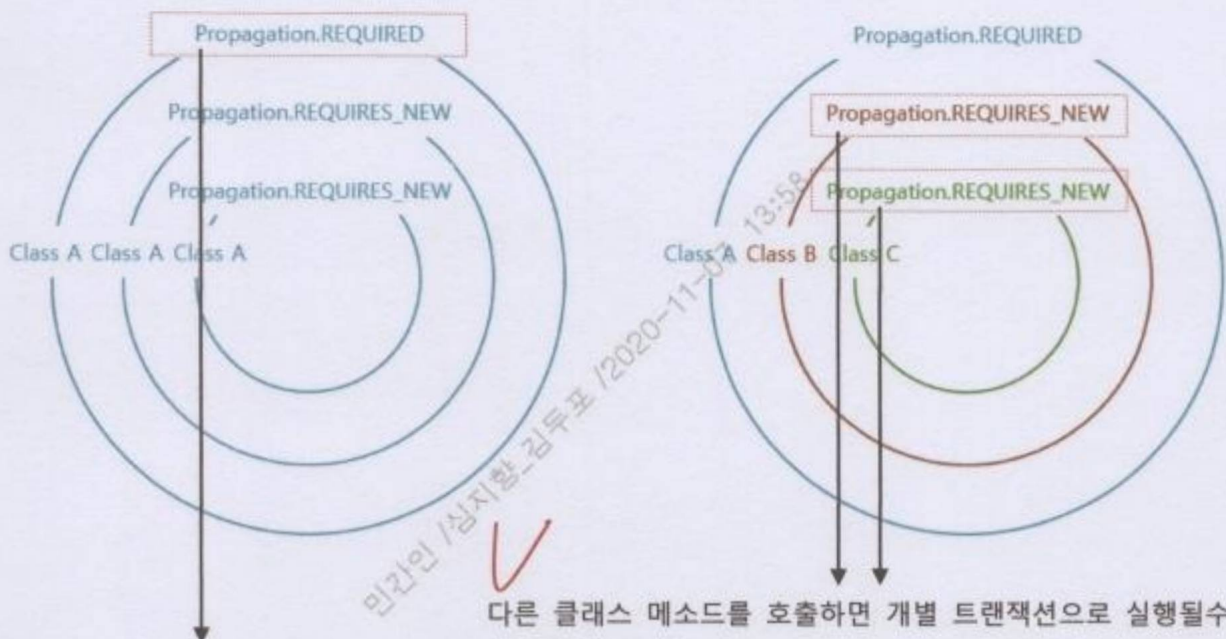
■ `@Transactional(propagation=Propagation.REQUIRED)`을 쓴다.

■ 트랜잭션이 필요없는 경우에는 `@Transactional` 을 쓰지 않는다.

■ 하나의 클래스 내에서는 최초 호출되는 메소드 이외에 나머지 메소드에 선언된 `Propagation` 속성은 아무 의미가 없다.

■ 단순 조회시에는 `@Transactional(readOnly=true)` 을 쓰는데 사실상 의미가 없다.

■ `Propagation.REQUIRED` 로 선언된 메소드가 다른 `Propagation.REQUIRED` 로 선언된 메소드를 호출하게 되면 하나의 트랜잭션으로 묶이는데 이때 해당 메소드의 트랜잭션을 별도로 빼고 싶으면 `Propagation.REQUIRES_NEW` 쓰는데 별도의 Class로 새로 작성해야 한다.



같은 클래스내에서 이면 제일 밖에 있는 메소드의 `Propagation`이 적용된다.

(3) 예제

```

package kr.gojrcSys.rb«.sevice;

import Java.util.HashMap;
import java.util.List ...

@>S«rv»ce
public class UserMgtService
{
    C^Autowired
    prhrste CommonDao daoc
    必 Auto—red
    private Log Service logSecvlcc

    必 T ran$»ct*onaH readOnly=»ue)
    public list<Map<String, Object>> sei«ctUserjt$(Map<String, Object* params)
    {
        return dao.queryForUst(*UserMgtseleaU5ertl$. params);
    }

    i^TranMctio i. i(prop»ga6on « PropagationREQUIKBO»
    public void s«veUs«ftist(List<Map<Str 1 ng, Obj<ct» htDala)
    {
        forfint i » 0; i < lsiDaUjizeO; i**)
        {
            If(^owStdus.equals(RowStatu$_kINSERT))
            {
                // leaveaLogfuserData)! ;
                JogSevice.teavealog(uMrrData);
                daainserCUUserMgtinsertUser*. userData);
            }
        }
    }

    @>Trinsactional(pf opagat »nc = Pt opagat »on. REQUIRE^. NEWt
    public void leaveaLog(Otojecl obj)
    {
        daainserfLogM9t.in$erUot)Log®, obj) ;
    }
}

```

```

public class LogSe«vice
{
    , 순' Tran^ction^^piopdgdttion=Propagation REQUIRES_NEW
    public void ieaveaLogfObject*o~|
    <
        d4O.lftMrtfU)gMgtinseatobLog.. objX
    }
}

```


사. Stored Procedure

(1) 개요

비즈니스 로직을 Stored Procedure에서 처리하고자 하는 경우의 예제

(2) 작성요령

Mybatis Query 작성시 statementType은 CALLABLE로, parameterType은 map으로,

이 2, input parameter/ output parameter 관련하여 각 jdbcType과 javaType에 맞게 선언하여 사용한다.

(3) 예제

① 단순 Procedure

- Return 이 없는 경우

```
<select id="test_proc1" statementType="CALLABLE"
      ( call test_proc1(#{ipt, mode-IN, jdbcType-VARCHAR, javaType=String}))
</select>

public void procedureCa111() throws Exception
{
    Map<String, Object> param = new HashMap<String, Object>();
    param.put("ipt", "2");
    dao.queryForObject("TestMgt.test_proc1", param);
}
```

② String Out Parameter Procedure

- Return이 있는 경우

```
<select id="test_proc2" statementType="CALLABLE" parameterType="map">
  { call test_proc2(
    #{ipt, mode-IN, jdbcType-VARCHAR, javaType=String},
    #{rst1, mode-OUT, jdbcType-VARCHAR, javaType=String},
    #{rst2, mode-OUT, jdbcType-VARCHAR, javaType=String})
  }
</select>

public void procedureCall2() throws Exception
{
    Map<String, Object> param = new HashMap<String, Object>();
    param.put("ipt", "2");
    dao.queryForObject("TestMgt.test_proc2", param);
    logger.debug("rst1: " + param.get("rst1") + ", rst2: " + param.get("rst2"));
}
```

③ Cursor Out Parameter Procedure

• Return이 Cursor인 경우

```
<resultMap id="testMap" type="java.util.List">
  <collection property="list" ofType="Map">
    <key column="id" type="String" />
    <key column="name" type="String" />
    <key column="age" type="Integer" />
  </collection>
</resultMap>

<select id="test_proc3" statementType="CALLABLE" parameterType="map">
  { call test_proc3(
    #{ipt, mode-IN, jdbcType=VARCHAR, javaType=String},
    #{rst, mode-OUT, jdbcType=CURSOR, javaType=ResultSet, resultMap=testMap})
  }
</select>

public void procedureCall3() throws Exception
{
    Map<String, Object> param = new HashMap<String, Object>();
    param.put("ipt", "2");
    dao.queryForList("TestMgt.test_proc3", param);
    Logger.debug("param: " + param);
}
```

\$ 트랜잭션

```

CREATE OR REPLACE procedure TEST.test_proc4
(ipt in varchar2)
is
begin
    -- Insert into test1 values((select max(seq) * 1 from test1)/ 919 ,
    92\ '3', 949);
    --insert Into testJ values((select max(seq) ♦ 1 from test1), •J 9,
    92\ 93\ 94 9);
    insert into test1 values(9, '1', '2', •3*, •4•);
    insert into test1 values(10, '1•, '2', '3', *4*);
    commit;
exception when others then
    rollback;
    raise;
end;
/

```

Default 로는 Stored Procedure 가 알아서 Transaction 을 관리 하고,

Spring 의 Transaction 0| 해당 Procedure 2| Transaction 을 관리할 수 없다,

단지, Procedure 로부터 Exception ft 받으려면 rollback; 다음에 raise;를 써줘야 한다.

아. Exception 처리

(1) 개요

즈'

■ Controller 를 시작으로 Service, DAO Business Logic 을 수행하는 도중에 Error 가 발생하게 되었을때, try/catch Block 으로 별도 제어하지 않는 한 Dispatcher Servlet 으로 에러가 전달되며, Dispatcher Servlet S Exception Advisor S. 예외를 던져 공릉단에서 이틀 수집하여 처리한다.

■ Exception 은 2 가지 Case 에서 발생한다. Checked Exception, Unchecked Exception.

| \$스ption 이 예상이 되느냐, 안되느냐의 차이.

■ Checked Exception 과 같은 경우 catch 구운에서 new SysException 으로 새로 예외메시지를 작성해서 처리한다.

r 실제 오류가 아니고, Business 상황의 오류인 경우싶은경우 new BizException 으로 예외메시지를 K해서 처리한다.

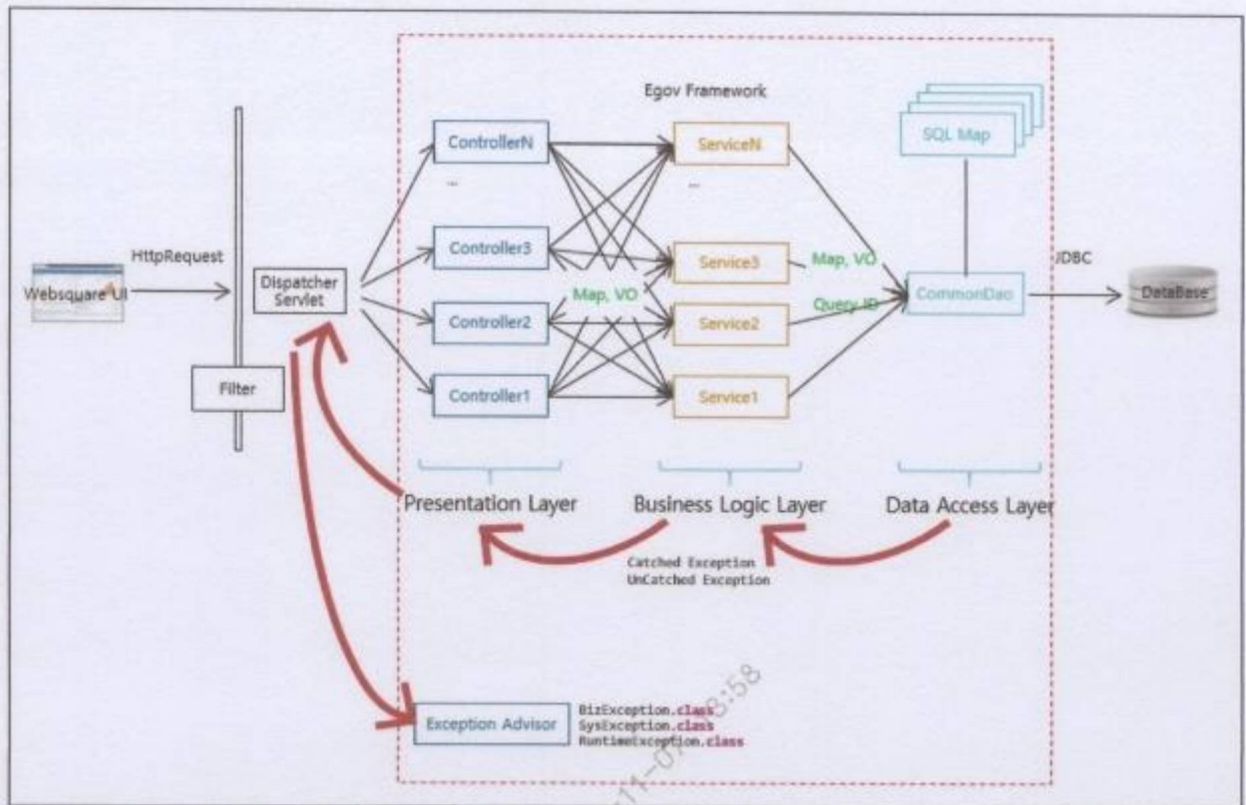
izException 2| 오류메시지는 msg.properties 를 이용하여 아래의 예와 같이 작성한다.

ex1. throw new BizException(MessageSourceAccessor.^ef^/ess<a^e<7nspCbd<?77>

■ 복잡한 메시지인 경우 그냥 한글로 작성하여 아래와 같이 던진다.

ex2. throw new BizException(numRow + ' ★ data1 + 7 에서 오류가 발생하였습니다.y,

■ NullPointerException 과 같은 예기치 못한 RuntimeException 의 경우 default S. '알수없는 오류가 발생하였습니다•의 메시지로 처리함.



(2) 예제

```

@Transactional(propagation=Propagation.REQUIRED)
public void test(List<Map<String, Object> gridData)
{
    File file = new File("c:/test.txt");
    try{
        file.createNewFile();
    }
    catch(IOException e)
    {
        throw new SysException(MessageSourceAccessor.getMessage("msgCode1"));
    }

    if(file.getName().equals("test.txt"))
    {
        ...
    }

    List list = dao.queryForList("UserMgmt.selectUserList", param);
    if(list.size() == 0)
    {
        throw new SysException(MessageSourceAccessor.getMessage("msgCode1"));
        // new SysException(MessageSourceAccessor.getMessage(numRow + " (" + data1 + ") 에서 오류가 발생하였습니다.");
    }
}

```

→ Checked Exception 처리 방법

→ UnCatched Exception
으로 "알수없는 오류가 발생하였습니다" 로 처리됨.

→ 업무상 예러 처리 방법

자. Properties 사용

(1) 개요

- 어플리케이션 공통적인 속성 또는 고정적인 속성 ■을 «도의 파일로 관리.
- System, Message 2 가지.
- 연계 URL 이라든가 로그 디렉토리 경로등 시스템 관련 속성들은 `sys.properties` 를 사용.
- 메시지는 `message.properties` 를 사용한다.
- 메시지는 전제공통 메시지 및 각 파트별 메시지로 구분하여 관리한다.

(2) 작성요령

key = value 쌍으로 입력한다.

① `sys.properties`

- 시스템 Properties

```
db.driverclassname = cubrid.jdbc.driver.CUBRIDDriver
db.url = jdbc:cubrid:192.168.0.201:30000:actsdb:::charset=utf-8

db.username = actsapp
db.password = actsapp
file.upload.dir = C:/DHRMISIDE/workspace/DHRMIS/WebContent/WEB-INF/files
file.upload.tmp.dir = C:/DHRMISIDE/workspace/DHRMIS/WebContent/WEB-INF/files/tmp
file.upload.threshold = C:/DHRMISIDE/workspace/DHRMIS/WebContent/WEB-INF/files/tmp
file.upload.max.size =2048576000
```

② `message.properties`

- 업무 메시지 Properties

```
#####
# 공통메세지
#####
error.cmn.unknownerror = 알수없는 오류가 발생하였습니다. 관리자에게 문의하여 주세요.
error.cmn.access.denied = 권한이 없습니다.

#####
# 인사메세지
#####

#####
# 평정메세지
#####

#####
# 자력메세지
#####
```


(3) 예제

1. Java에서 Value Annotation으로 사용

```

public class CommonDao {
    @Value("${sysProps.file.upload.path}")
    private String uploadPath;

    public void downloadXifFile<String statement, Map<String, String> param, String filename, HttpServletResponse res) throws IOException {
        // ...
        String tmpFileName = UUID.randomUUID().toString();
        File tmpFile = new File(uploadPath + "/" + tmpFileName);
        FileOutputStream fos = new FileOutputStream(tmpFile);
        write(fos);
    }
}

```

상단 주석: @Value annotation 사용하여 Properties 값을 가져올 수 있다.

2. Java에서 MessageAccessor Util을 이용하여 사용

```

@ControllerAdvice
public class ExceptionHandler {

    @Autowired
    private MessageAccessor messageAccessor;

    @ExceptionHandler({ BizException.class })
    public void bizError( HttpServletRequest req, HttpServletResponse res, BizException e,
        HttpServletRequest request, HttpServletResponse response) throws IOException {
        String message = messageAccessor.getMessage(e.getMessage());
    }
}

```

3. Spring Configuration Xml File에서 사용

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans.xsd">

    <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"
        p:driverClassName="${sysProps.db.driverClassName}"
        p:url="${sysProps.db.url}"
        p:username="${sysProps.db.username}"
        p:password="${sysProps.db.password}" />

</beans>

```

차. Session 처리

(1) 개요

SpringSecurity2| UserDetails를 확장한 UserInfoHolder의 Static method 사용
개발 요구사항에 맞게 필드 추가 예정.

(2) API

Method	설 명
static UserDetails getUserInfoO	세션에서 사용자객체 가져온다.
static String getUserIdO	세션에서 사용자의 id 가져온다
static String getUserNmO	세션에서 사용자의 이름 가져온다.
static String getDeptCd O	세션에서 사용자의 부서코드 가져온다.
static String getDeptNmO	세션에서 사용자의 부서명 가져온다.
-	...

(3) 예 제

1. Java단에서 쓸때

```

@Service
public class TestService
{
    ...
    @Transactional(propagation=Propagation.REQUIRED)
    public void calculateSomeLogic(Mip param)
    {
        ...
        logger.debug("사용자아이디 : " + UsermfoHolder.getUserIdOK);
        ...
    }
}

```

2. MyBatis단에서 쓸때

```

<select id="selectUiefRoleMipLsrparameterType" resultType="HashMap">
    select decode(T1.user_id, '0', T) chk,
           nvl(T1.user_id, '{user1d}') user, id,
           T2.role_cd,
           T2.role_nm,
           T2.role_desc
    from T_USCR_ROLE_MIPNG T1,
         T_ROLE_MGMT T2
    where T1.role_cd(+) = T2.role_cd
           and T1.user_id(+) = (SS.USER.ID)
    order by T2.role_cd

```


카. File 처리

(1) 사전지식

1. 파일 테이블 구조는 아래와 같고, 업무데이터베이스에서는 파일 마스터 테이블의 PK 인 **atch_filejd** 만 가져간다

« 무테이저

OHU 마스터

파빈 상체

perwnJd	pmorcrwne	•tdiffWd	atch.WeJd	file_desc	atdOHeJd	하•제	Me.rwne	file_path
1234	홍길동	1	1234		1234	1	test1.pdf	/u01/fileV202Q/07/29Ae»t1 pdf
					123	2	tet2-pdf	/u01/fil»»/202(y07/29Aest2.pdf

2. **atch.filejd** 는 Hashing 함수로 생성된 값을 사용한다.

한 예로, <http://www.dhrmis.mil/file/download.do?atchFileId:1 &fileSeq=1>

의 주소를 발견하게 된다면

, 스

<http://www.dhrmis.mil/file/download.do?atchFileId=1&fileSeq=2> 와 같이

다른 **seq** 번호를 유추할수 있어 다른 사람의 데이터도 볼수 있게 된다.

sequence 번호를 유추할수 없게 난수로 표현하는 방법을 사용.

Hash 함수 **SHA-2 Series** 중 하나인 **SHA-256** 을 사용.

이 알고리즘으로 Hashing 하면 16진수 표기 법에 따라 64 자리 난수가 리 턴.

참고로 중복발생 가능성은 $1 / 2^{256}$ 로 사실상 불가능 하다.

1

/

1579208923731619452359850008689078523233405603945784007913129639936423570985611579208923731

19452359850008689078523233405603945784007913129639936

정도 (78 자리)

예제는 B94D27B9934D3E08A52E52D7DA7DDABFAC484EFE37A5380EE9088F7ACE2EFCD9 와 같다.

(2) 첨부파일 처리

① 첨부파일 업로드 생물

■ 파일 업로드 UI 컴포넌트를 이용하는 경우

• 파일업로드폼을 넣고자 하는 위치에 <div : ▶생성하고, 해당 div id 를 parameter 로 com.file.multiForm 호출 simpleform

```
<body ev:onpageload«scwin.onpageLoad" ev:onpageunload-so , t/».on/x7ge4intoO 方テ
  <div id«test1 * style^'i^idth:85epx;height:161px; #X/div>|
</body>
```

```
scwin.onpageload - function()
```

```
{
  com.file.simpleForm('test1', '10', '1',
    function()
    {
      console.log( •■어 重드뵈고 臺기한 하고 실은 오드醫 업어주세요. ');
    }
  )
  function(response)
  {
    console.log( •과@이 업麗드 f고 나서 오향말 玉드醫 石어주세요. •시
    console.log(response['list.][0].fileId);
    console.log(response['list*](0).fileSn);
  }
}
};
```

파일업로드 (simple) x

M0214

파일선택 2. OZ 구성도.pptx 삭제

multiform

```
<body ev:onpageload="scwin.onpageload" ev:onpageunload="scwin.onpageunload">
  <div id='test1' style='width:850px;height:161px;'></div>
  <!--xf:group style="width:850px;height:161px;" id="test1" tagname="div">
</body>
```

```
scwin.onpageload = function()
```

```
{
```

```
  com.file.multiForm('test1', '',
    function()
    {
      console.log('폼이 로드되고 초기화 하고 싶은 코드를 넣어주세요. ');
    },
    function(response)
    {
      console.log('파일이 업로드 되고 나서 실행할 코드를 넣어주세요. ');
      // JSON.parse(response).list[0].atchFileId
      // JSON.parse(response).list[0].fileSn
    }
  );
};
```

파일업로드 (multi)					
M0062					
[파일선택]					
파일명	용량	등록자	등록일	삭제	
파일을 드래그하여 첨부할 수 있습니다.					
2. OZ 구성도.pptx 75.4 kB					

staticform

```

<body ev: onpage load* "sctfin. onpage Load" ev: onpageunload* "sciffin. onptigeunLood^y
  <div id= *test3* style=tridth:85€px;height: 161px; *X/div>
</body>

```

```

scwin.onpageload ■ function()
{
  v<r fileinfo ■'title': [[ 테스트il{"title": •테스量:r}]]{'title•: •테스 트3'}}];

  co«•file•staticFoni(*test3*, fileinfo, •
    function()
    {
      console, log(•이 量드S고 違기ft 하고 쇼은 玉드蓐 호어주세a*' 시
    },
    f function ( response)
    {
      console.log(•과일어 Si드 되고 나서 실«할 교드5 양어주세요, ');
      console • log( response [ ' list • ] [e] .fileId) ;
      console.log(response[•list*][0].fileSn);
    }
  );
};

```

파일업로드 (static) x

M02H

at 파일선택 테스트 1

1 < 확tig 액 q 스트2 [z]

파일선택 M 스트3

저장

■ 직접 파일 업로드 醫 합 경우

- Multipart ContentType 으로 (<form method="post" enctype="multipart/form-data">...</form>)

Form 을 Submit 하고, Controller 에서 아래와 같이 파일 정보를 받아서 처리.

DB 에 등록할지는 업무의 성격에 맞게 설계

```

@RequestMapping("/rbi/saveUserlist.do")
public void saveUserlist(MultipartRequest req, HttpServletResponse res) throws Exception {
    // get parameter
    req.getFileUpload();
    List<File> fileVOs = req.getFile();

    // get file information
    for(int i = 0; i < fileVOs.size(); i++) {
        System.out.println(fileVOs.get(i).getFileName());
        System.out.println(fileVOs.get(i).getOriginalFileName());
        System.out.println(fileVOs.get(i).getFileSize());
    }
}

```

민간인 /심지환-김두포 /2020-11-07 13:58

② 첨부파일 다운로드 샘플

-atchFileId 와 fileSeq 를 넘겨 다운로드

```
scwin.btnDownload onclick : function(e)
{
    com.file.download(1, 2);
};
```

민간인 /심지형-김두포 /2020-11-07 13:58

(3)

(4).엑셀파일 처리

① Grid 엑셀파일 업로드

```
var gridId ■ "grdUser**";  
var type ■ -xlsx";  
var options » { fileName : "test.xlsx" };  
  
com.gridDataUpload(grdObj> type, options);
```

② Grid 역셀파일 다운로드

```
var gridId ■ "grdUser";  
var infoArr = {}; // rtbArr 그리드에 대한 내용을 추가로 다른 셀에 표현하는 경우 사용하는 배열  
var options « { fileName : "test.xlsx"  
};  
  
com.gridDataDownload(gridId, options, infoArr);
```

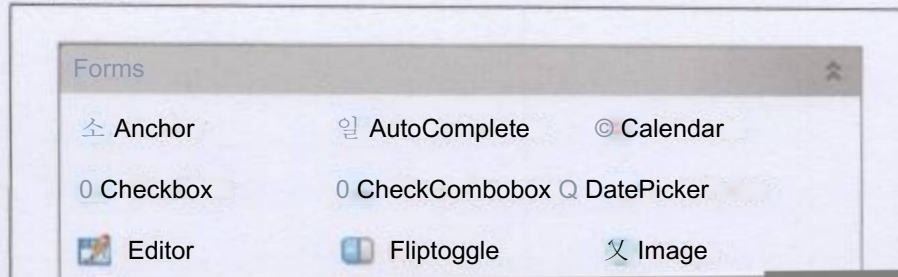
於

민간인 /심지함-김두포 /2020-11-07 -58

⑧ DB 엑셀파일 업로드 (csv, xk xlsx)

■ 엑셀파일을 업로드하여 다단에 Insert하는 기능

- ui source



- script source

Upload 컴포넌트 이용하고, 속성에서 id와 action 부여한 다음 아래와 같이 특정한 버릇을 클릭했을때 스크립트 처리

```
scwin.btnSubfnit onclick ■ function(e)
{
    uploadl.subnit();
};
```

- server source

Controller에서 file 정보 받아서 CommonDao 호출

```
^RequestMappingC/rba/saveUserlist.do**)
public void saveUserList(MciRequest req, Model model) throws Exception
{
    // get parameter
    req.fileUpload();
    List<FileVO> fileVOs - req.getFile();

    // get file infoHMTL.on
    System.out.println(fileVOs.get(6).getFileStreCours());
}
```

CommonDao 의

A

```
public void uploadCsvFile(String sqlld, Class cis, String fileName, String cvsSplitBy)
```

```
public void uploadXlsFlie(String sqlld, String fileName, RowConverter rowConverter)
```

```
public void uploadXlsxFlie(String sqlld, String fileName, RowConverter rowConverter) 이 용
```

```

^Override
@Transactional(propagation = Propagation.REQUIRED)
public void uploadCsvTest(FileVO fileVO) throws Exception
{
    dao.uploadCsvFile("**TestMgt.insertTest2", fileVO.getFileStreCours(), ",");
}

^Override
@Transactional(propagation = Propagation.REQUIRED)
public void uploadXlsTest(FileVO fileVO) throws Exception
{
    RowConverter<TestVO> roConverter = new RowConverter<TestVO>()
    {
        ^Override
        public TestVO convert(Object[] row)
        {
            return new TestVO((Object)row[0], (Object)row[1], (Object)row[2], (Object)row[3]);
        }
    };
    dao.uplcNidXlsFile("TestMgt.insertTest3", fileVO.getFileStreCours(), roConverter);
}

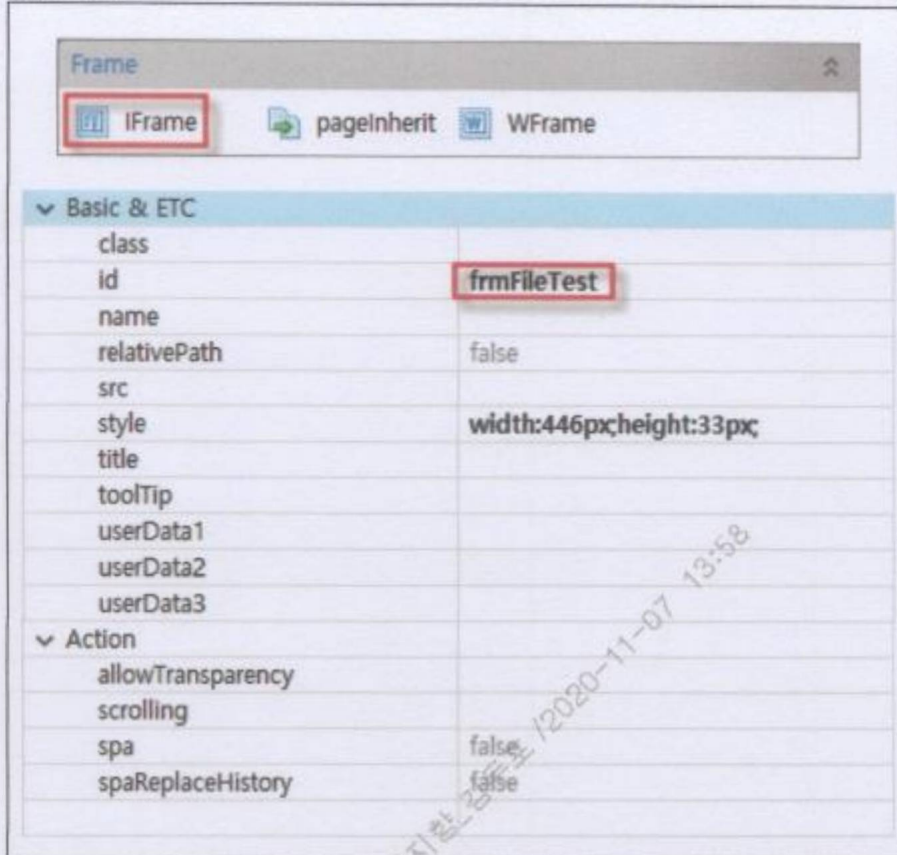
^Override
@Transactional(propagation = Propagation.REQUIRED)
public void uploadXlsxTest(FileVO fileVO) throws Exception
{
    RowConverter<TestVO> rowConverter = new RowConverter<TestVO>()
    {
        ^Override
        public TestVO convert (Object[] row)
        {
            return new TestVO((String)row[0], (String)row[1], (String)row[2], (String)row[3]);
        }
    };
    dao.uploadXlsxFile("TestMgt.insertTestB", fileVO.getFileStreCours(), rowConverter);
}

```


④ DB 엑셀파일 다운로드 (csv , xls, xlsx)

■ 쿼리콜 던져 엑셀파일을 Download하는 기능

- ui source



IFrame 컴포넌트 이용하고, 속성에서 id를 부여한 다음 아래와 같은 스크립트 처리

```
scwin.btnCsvDownload.onclick = function(e)
<
    frmFileTest.setSrc(* Ztest/downloadCsvTest.do *);
    ;
```

- server source

CommonDao 의

```
public void downloadCsvFile(String statement, Map<String, Object> params, String fileName,
, HttpServletResponse res)
public void downloadXlsFile(String statement Map<String, Object> params, String fileName,
, HttpServletResponse res)
public void downloadXlsxFile(String statement, Map<String, Object> params, String fileName,
, HttpServletResponse res) 이용
```

```

@Override
public void downloadCsvTest(HttpServletResponse res) throws Exception
{
    Map<String, Object> params = new HashMap<String, Object>();
    params.put("srchTxt", "aaa");

    dao.downloadCsvFileWithRowHandler("TestMgt.selectCsvDownloadTest", params, "test-csv.csv", res);
}

@Override
public void downloadXlsTest(HttpServletResponse res) throws Exception
{
    Map<String, Object> params = new HashMap<String, Object>();
    params.put("srchTxt", "aaa");
    dao.downloadXlsFileWithRowHandler("TestMgt.selectXlsDownloadTest", params, "test-xls.xls", res);
}

@Override
public void downloadXlsxTest(HttpServletResponse res) throws Exception
{
    Map<String, Object> params = new HashMap<String, Object>();
    params.put("srchTxt", "a");

    dao.downloadXlsxFileWithRowHandler("TestMgt.selectXlsxDownloadTest", params, "test-xlsx.xlsx", res);
}

```

洙 /ui/test/largeFileTestaml 참고

[J WebSquare] x +

分 구 ① 1220.0.1 公 080/websquare.do?w2xPath=/sys/sample/pub/main.xml

 국방인사정보체계 Q

1. 퍼리싱

1. 개발예제

- 0 휴크리
- 그리드머舊
- 디1품9과일

Hit

- 파일업로드
- 역舊-0 유 01%
- 역舊 관리
- 르石머션 머스s
- 사쯤자 조려
- 사물자•역If 0|«

라 업*드 x 대응항과업 x

to 다운fifi csv download (1000건)

Xb 다운 생드 xls download (1000건)

Xlsx 업로드 Mhx downk»d (102*20)

Cw 업로드 皇 submit (이 파일 S 轉스重화M세fi gj >

x)s*S 드 皇 I subnw (이 파터스B 轉4요 B)

Xlu2 로드 A subnwr (01 파일E 轉스轉보세fi H)

장 한셀 지원범위 합의 필요

Hancell 은 2007 년까지 .nxi 넥셀 포맷으로 서비스, 데이터 제한이 65535 건.

이후 .cell 포맷이 나오면서 메모리가 허용하는 범위내에서 무제한 데이터 저장가능.

xls 와 xlsx 의 차이와 같음.

2007 년 이후 .cell 포맷이나 .xlsx 나 둘다 OpenXML 기반의 엑셀 ! 프로그램이 제작되어 같은 Platform 이라 어느정도 호환은 되나 기반이 같은것이지 똑같은건 아니라서 호환의 범위가 좁다.

그 범위에 대한 설명.

서버에서 xls 포맷으로 다운로드 받아 .cell 로 저장하면 정상 한셀로 작동.

서버에서 xlsx 포맷으로 다운로드 받아 .cell 로 저장하면 정상작동 안함.

결론 =65535 이상 건수의 데이터에 대해서는 무조건 xlsxfi 사용한다.

65535 이하 건수의 데이터에 대해서는 J(1s, .cell 둘다 지원한다.

한컴오피스에서 .cell 에 대한 download Java API 를 제공한다면 해결이 되겠지만. 그러하지 않다.

於

타. Paging 처리

(1) 개요

Java단은 전자정부 Paging Utility# 사용하여 Paging을 계산하고,

Query단에선 해당 Data를 찾기 위한 Cost를 최소한으로 줄이기 위해 RK만 먼저 조회하는 Temp Table을 선언하는 방식을 사용하다.

(2) 예제

1. Controller

```

import egovframework.rte.pci.mvc.tag$.uLpagination PagmMloninfQ ;

^RequestMapping/setectUserPa^natedList^
public void $lectUserPaginatedList(MciRequ«t req, Model mode" throws ExceptorM
// ge< parameterM-
Map<String, Object> pvanw ] req.getParam("dsPar>ml;

// set pagint parameter
^nt currentPageNo = 1; // 현 지폐 이 지
int recordCount = 10c // 한산면에 보여> 흑黑 1수
<Kpanmj.getC - cufTentPageNol l» null) currentPageMo : integer MM/eO4params.getCcurrentPdgeNo")◆■□;
^Kparams->getCreordCount") l» nuN ) recordCount = Integer.짜/eO<pa , an lgeU.rwo , <Kou"t)◆■□;
Paginationinfo paginatio^Info • new PaginationInfoO ;
paginationinfo->setCurrentPageNofcurrentPageNo);
pagntatlon»nfo.setRecordCountPefP*ge<reco«jCounX
pv»m$.put("fif5UrMde)c' pagination!n(age<FirstRecordIndexOX
params->putRecordCounf, pagination!Info.getRecordCoun(Pe<P>g«0) ;

// invoke service method
Ust<?> result = userM9mtService«J*<ectUserPaginatedLf$<t<pafams);

//response
modeLaddAttributerdsUseflitt , rtsutU

```

2. MyBatis

```

Mlth temp ti
(select •
  from (Ml▶▶ct paging.4,
        rownum as rnuii,
        count(1) over() as total_count
        from (itlcct user^id. userna.
              fro, T_US«_NGM7 where 1-1
              sdfasd
              order by user_id d«sc) paging)
  where rnua <1(CDATA(>)]> 히fir&tIndex»
        and rnui , <[COAIA[<]]> (S{firstIndex) ◆ f(recordCount)))
&eltct Tl.total.count,
        Al.user.Id
        Al.user.raai,

        fro» T-USER_MGMT AI,
        THIP 11
        where Al.user_id - Tl.user^Id
order by Tl.mum

<ielect id" s^lectUserPjffinj^edlis" parameterOasss "tuuhmjp" resultCU^s"
select IW5E.UU1,

```



```

T1.emplyr_id,
T1.user_nm
from COMTNEMPLYRINFO T1,
(select *
  from (select PAGING.*, rownum as rnum, count(1) over() as total
        from (select emplyr_id,
                     orgnzt_id,
                     user_nm,
                     passwd,
                     empl_no,
                     ihidnum
                  from COMTNEMPLYRINFO
                  order by emplyr_id desc) PAGING)
  where rnum <![CDATA[>]]> ${firstIndex}
        and rnum <![CDATA[<=]]> (${firstIndex} + ${recordCount})) PAGE
where T1.emplyr_id = PAGE.emplyr_id
order by PAGE.rnum
</select>

<select id="selectUserPaginated2List" parameterClass="hashmap" resultClass="camelCaseMap">
select PAGE.total,
      T1.emplyr_id,
      T1.orgnzt_id,
      T1.user_nm,
      T1.passwd,
      T1.empl_no,
      T1.ihidnum
      to_char(T1.sbscrb_de,'YYYY-MM-DD') as sbscrb_de
from COMTNEMPLYRINFO T1
<include refid="fw.sqlmap.CommonSql.pagingBegin"/>
      select emplyr_id from COMTNEMPLYRINFO order by emplyr_id desc
<include refid="fw.sqlmap.CommonSql.pagingEnd"/>
where T1.emplyr_id = PAGE.emplyr_id
order by PAGE.rnum
</select>

```

파. 각종 Data 처리

(1) XML Parsing

1. DOM Parsing

• -9. TestCase- 창조

2. SAX Parsing

- -9. TestCase* 창조

(2) JSON Parsing

- M9. TestCase' 참조

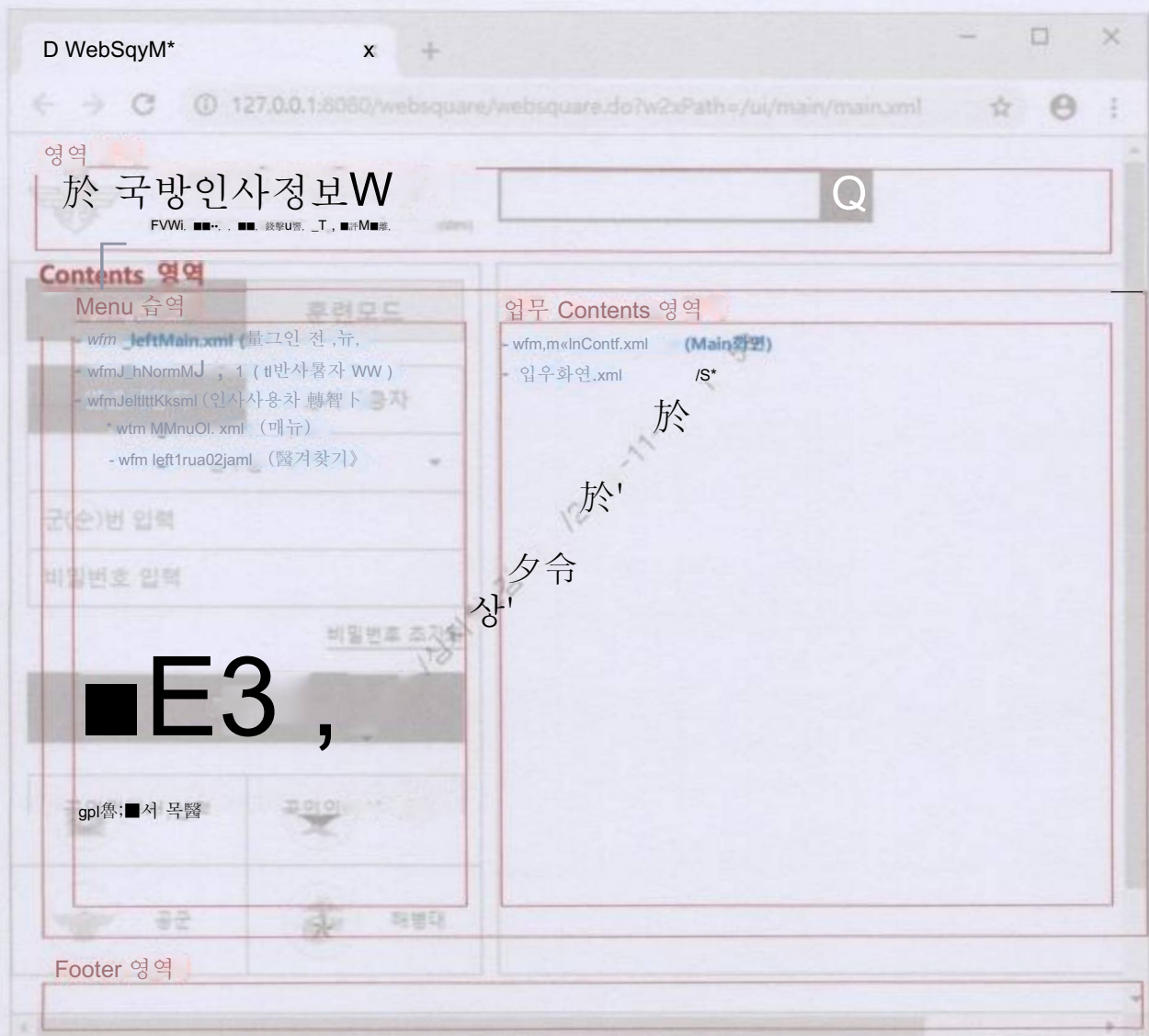
민관인 /심지환-김두포 /2020-11-07 13:58

6.2 UI 프로그램 (Websquare) 개발

가. Frame Set 개요

1. Page는 기본적으로 Top영역과, Contents, Footer 3가지로 구분하고 Contents안에는 다시 Left Menu와 업무Contents 영역으로 구분한다.

개발과 관련된 실제 영역은 업무Contents 영역이다.



전체 너비는 1920 이고, Left Menu 는 300 외, Padding 10 몰 제외하면 업무영역은 16W 내에서 개발

2. 사용자별 Left Menu

■ 로그인 페이지

실제모드	훈련모드
일반사용자	인사사용자
군구분 ▼	
군(순)번 입력	
비밀번호 입력	
비 a 번호 초기화	
로그인	
공인인증서 신청	공인인증서 용록
공인인증서 로그인	

■ 일반사용자 페이지

1 개인자력	攻'	>
1 보직신청		>
1 지원업무		>

■ 인사사용자 페이지

권한 S 매뉴	클거찾기
1 인사명령 (상훈)	>
1 제계관리	>

나. 코딩 개요

개발자의 생산성 및 이후 유지보수를 위한 가독성 《readability》를 고려하여

아래와 같이 변수선언, 화면초기화, 이벤트함수, 트랜잭션함수, 사용자정의함수 순으로 작성한다.

```

"
// 초기화
//.....
scwin.on(goto4d) = function()
(
    scwin.fnSearch();
);

//.....
// Event 함수
//.....
scwin.btnSearch_ofclick = function()
<
    scwin.fnSearch();
>;

scwin.btnSave.onclick = function()
(
    scwin.fnSaveList();
);

//.....
// Transaction 함수
"
scwin.fnSearch - function()
{
    var searchoption = {
        id      : 'fnSearchList',
        action  : '/rba/selectUserList.do',
        ref     : ['dmParam'],
        target  : (fid : -dlUserList, -key : -userList5f)
        // target : ['id_d'] UserList -key ~ rList
    };

    com.executeSubmit(searchoption); DataSet의 이름이 서버와 내가 다를 경우는
};

scwin.fnSave = function()
{
    var saveOption = {
        id      : 'fnSaveList',
        action  : '/rba/saveUserList.do',
        ref     : ['dlUserList'],
    };

    com.executeSubmission(saveOption);
};

//.....
// Callback 함수
"
scwin.fnCallbackHandler = function(options)
{
    var _callbackid = options.h.replace(/p.id/, '*');

    if(_callbackid == 'fnSearchList')
    <
        console.log('Local callback...');
    }
    else if (_callbackid == 'fnSaveList')
    {
        scwin.fnSearch();
    }
};

```


SB DataCollection Editor Dialog

dlUserlist • dataUst 신규 v ID 변경 Type 변경

Column Data

네 5

No	id	name	dataType
1	emplrId	name1	text
2	userNm	name2	text
3	passwd	name3	text
4	emplNo	name4	text
5	ihidnum	name5	text
6	birthdy	name6	text
7	fxnum	name7	text
8	houseAdres	name8	text
9	areaNo	name9	text
10	zip	name10	text
11	emailAdres	name11	text
12	pstinstCode	name12	text
13	esnlId	name13	text
14	crtfcDnValue	name14	text

Apply and Close Apply Cancel

라. 기본 공통 Script

- Websquare Page 에서 사용할 /websquare/configcml 에 설정된 Default Javascript 3 가지
gcm.js, com.js, valid.js 중에서 주요한 함수들만 설 명.

《1》 트랜잭션 함수

DataBase에 실제 CRUD* 하게되는 Websquare Adapter와 인터페이스를 하게 되는 함수.

예제.

```
var searchoption = {
    id      : 'fnSearch',
    action : '* /rba/selectMenuList.do',
    ref : [dsParan, ''],
    target : ['*menuList*']
};
com.executesubmission(searchoption);
```

(2) Callback 함수 처리

com.executeSubmission 트랜잭션이 실행될 때

1. 트랜잭션 직전 global 로 pre-submission 함수가 실행된다.
2. 트랜잭션이 정상적으로 끝나고 나면 global callback 함수가 실행되고,
local 에 선언된 callback 함수가 있으면 그 다음 순차적으로 실행된다.
3. 트랜잭션에 오류가 발생하면 global error-callback 함수가 실행되고,
local에 선언된 Callback함수가 있으면 그 다음 순차적으로 실행된다.

(3) Popup

팝업은 LayerPopup을 사용하며 종류에 따라 Alert Info, Confirm, Error로 구분한다.

팝업화면에서 확인버튼이나 취소버튼등을 누를 때 호출되는 Callback함수에 대해 API확인 필요.

(4) Validation

(5) Grid 관련 Utility

Grid Data Download

ex)

```
var options = { filename : 'test.xlsx' };
var infoArr = [];
com.gridDataDownload(grdUser, options, infoArr);
```

→ Grid Id , Excel 종류 (CSV, xls, xlsx)

Grid Data Upload

ex)

```
var options = { };
com.gridDataUpload(grdUser, 'xlsx', options);
```

→ Grid 化 Excel 종류 (CSV, 빈칸 xlsx)

(6) 공통코드

해당 업무화면에서 최초 1회 호출할 때 Transaction을 타고, 이후부터는 SessionStorage에 저장해 놓은 값을 사용.

ex)

```
var codeDatas = com.getCodeData(((id : -COM00001-), (id:"C0Mee@e2**})); > 공통코드 IdS
co«.setCowboBox(- rdoCode!, codeD경tas,CO"eewi); > 코드
co«.setco t>oBox(.a"bcc>de2.i codeoSus.G'eeeiK > 方, 동고 가 H정, 월 티ement Id
```

(7) 첨부파일

File Upload Form

ex)

```
<?xml-stylesheet href="/css//ile.css" type="text/css"?>
<script src="/js/ftLe/require.js" scopeExternal="true" scopeVariable="Mctt"/>
<script src="/js/ca«ofj.js" scopeExternal="true" scopeVariable="Mctt"/>
<script>
    Fn.File.nultiForiR('test 1 * , '1*,
        function()
        {
            console, log( • 용이 置터고 $기작 하고 싶은 코드醫성어주<요. • ) ]
        },
        function(response)
        {
            console.log(•파안이 업로드 되고 나서 일을< 코드■ 넣어주세요..)j
            " 3S0N.parse(response).list[6].filXd);
        }
    );
</script>
```

a 파일선택

파a音 드려그하어 점부<

파일명	용량	등* 자	등* 일	삭지
7z1805-x64.exe	undefined	undefined	undefined	삭지
ezPDFEditor.PersonaLSetup.3,0 3-4.exe	undefined	undefined	undefined	삭지

(8) Fusion Chart

Fusion Chart API를 참고. (WebSquare5 (SP3) 사용자 가이드4)df - 164 Page)

마. 전체 공릉 Script

(1) gcm.js

- 메인화면 《main.xml》 최초 로딩시 Object 1 개만 생성하여 사용
- 메인화면에서만 사용할 스크립트로, 업무화면에서는 접근 불가.

(2) comjs

- 각 업무화면 임로 개별 Object 생성. (대부분 WFrame관련)
- 모든 업무화면에서 공통적으로 사용될 유틸리티 등을 선언.
(MDI, Submission, Grid, 공통코드, Popup, Data , Event 및 기타 함수음)

① MDI 관련

```
// main.xml 의 windowcontainer 에 메뉴블 open 한단/
com.openMenu ■ function(psMenuId, poParam)

// main.xml 의 컨테츠영역의 화면을 이동시린匕卜
cofli.changeMenu « function(psTitle, psSrc, poParam)

// windowcontainer 에 Tab 읍 ad건 한다.
ccxn.addMindow « function(poTabObj, poOpts, poParam)

// 메뉴번호에 해당하는 메뉴정보를 return
com.getMenuInfo ■ function(psMenuId)

// 현재 화면의 웹스퀘어 page 경로를 반환한다.
com.getPageUrl « function()

// 실행 프레임 정보 조회
com.getActiveWindowInfo = function()

// tatframe 의 src 를 변경한다.
com.wfmSetSrc « function(poWframe, psSrc, poParam)
```


② 트랜잭션 관련

```
// 트랜잭션 실행
com.executesubmission ■ function(options, requestData, obj)

// Submission 객체 동적으로 생성한다.
com.createSubmission = function(options)
```

③ Grid 및 DataSet 관련

```
//GridView 와 바인딩된 DataList 객체 & 반환한다.
com.getGridViewOataList « function(gridViewObj)

✓/특정 위포노1트에 바인된 DataList나 DataMap의 정보를 반환한다.
com.getDataCollection = function(comObj)

// 데이터 리스S 필터 설정 함수.
com.dataListFilter = function(_dataCofnp, ^condition, ^append, options)

// 특정 컴포넌트에 바인된 DataList나 DataMap의 것범 이류올 반환한다.
com.getColumnname = function(comObj)

// DataCollection 객체의 변경된 에이터가 있는지 심사한다.
com.checkModified ■ function(dcObj Arr)

// Gridview Row 다제유 위한 CheckBox 동작■폭: 세팅한다.
com.setGridViewDelCheckBox ■ function(gridViewObjArr)

// dataList에 row. 추가한다.
com.insertRow = function(poIObj)

// dataList의 row ; * 시■제한다.
com.deleteRow « function (poDIObj, pbOelAHSatus)

// 그리드 엑셀 다운로드 fCSV, EXCELS 다운보드 받다.)
// downLoadCSV 나 downLoadExcel> S술
com.gridDataDownload ■ function(grdObj, options, infoArr)

// 그리드 엑셀 업로드
// uploadCSV 나 uploadExcel^ 호출
com.gridDataUpload = function(grdObj, type, options)
```

④ 공통코드 관련

```
com.getCodeData ■ function(params)

com.setComboBox ■ function(elld, datas)
```

⑤ 팝업 관련

```
com.alert = function<popupId, messageStr, userData, closeCallbackFncName>

com.confirm = function(popupId, messageStr, userData, closeCallbackFncName)

com.error « function(popupId, messageStr^ userData, closeCallbackFncName)

com.info ■ function(popupId, messageStr, userData, closeCallbackFncName)
```

```
// 팝업창을 연다.
com.openPopup = function(url, opt, data)

// 박업상을 닫는다. callbackFnStr을 이용하여 부모창의 callback함수를 호출한다.
com.closePopup = function(poRetObj, poEtcInfo)
```

◎ 데이터 관련

```
" 그림안에 포함된 컴포넌트의 입력 값에 대한 유효성검사를 한다.
com.validateGroup = function(grpObj, valInfoArr, tacObj, tabid)

// Gridview를 통하여 입력된 데이터에 대해서 유효성을 검증한다.
com.validateGridView = function(gridViewObj, valInfoArr, tacObj, tabid)

// 날짜유효성체크(yyyyMMdd 8자리) 시과이 return아니.
com.regExpDateFunc = function(param)

// 문자열을 날짜포맷으로 변경
com.strToDateFormat = function(psData)

// 일반전화번호 유효성 체크 후 영태에 맞게 윤_차"변경
com.telChkFormat = function(psData)

// 휴대전화번호 유효성 체크 후 정태에 맞게 "을 넣어서 변경
com.moblChkFormat = function(psData)

// 메일주소 체크한다.
com.checkEmail = function(str)

// 숫자 3자리마다 "/"표시 후 return
com.toFormatNum = function(psStr, pnDecimalLength)

// 새번대사리마다 감미 技시, 금래, setDi splay, ###8#46##0",
Mfn_userFormatter2" 상M된 str에 포매터를 적용하여 반환하다.
com.transComma = function(valuej type)

// dataCollection의 유효성 제? i를 수행한다.
com.checkValid = function(poOpts)

// 내외국인 주민등록번호 유효성검사를 한다.
com.checkPersonID = function(str)

// 사업자번호 유효성검사를 한다.
com.checkBizID = function(str)

// 법인등록번호 유효성검사를 한다.
com.checkCorpID = function(str)

// 주민번호 문자열에 Formatter(#####) 적용하여 반환한다.
com.transIdNum = function(str)

// 전화번호, setDisplayFormat("### t### - -####") - 입력된 str에 포매터를 적용하여
반환한다.
[어n.transPhone » function(str)
```



```

// 시간 - 입력된 String 또는 Number에 포메터들 적용하여 반환한다.
com.transTime ■ function(value)

// 여스트 - 입력된 str에 포메터들 적용하여 반환한다.
com.transText ■ function(str)

// 날짜 - 입력된 str에 포메터들 적용하여 반환한다.
com.transDate = function(str, type)

// 문자열 왼쪽에 일정길이(maxLen) 만큼 '0'으로 채우기
com.fillZero « function(str, maxLen)

// JSON Object로 변환해서 반환한다.
com.getJSON ■ function(str)

// JSON Object인지 여부를 검사한다.
com.isJSON » function(jsonObj)

// XML Document 객체인지 여부를 검사한다.
com.isXmlDoc ■ function(data)

// Object가 null, undefined^ "", {}, [] 일: 경우 true 이외의 경우는 false
return 한다.
com.isEmpty ■ function(poObj)

// Object가 null, undefined, "", {}, [] 일 경우 false 이외의 경우는 true^
return 한다.
com.isNotEmpty = function(poObj)

// Object가 null, undefined/ {}, []^ 경우 false | Object일 return#다.
com.ifEmpty = function(poObj, poConvObj)

// JSON 객체를 String 타입으로 반환한다.
com.strSerialize « function(object)

```

⑦ 기타

```

// 해당 그룹 안의 컴포넌트에서 엔터키가 발생하면 해당 컴포넌트의 값 DataCollect에
저장하고 objFunc 함수를 실행한다.
com.setEnterKeyEvent = function(grpObj, objFunc)

// 파라미터를 읽어 온다.
com.getParameter ■ function(paramKey)

// contextRoot 가 포함된 path문 반환한다.
com.getFullPath ■ function(path)

```

X Websquare API 는 아래 링크 참고

<http://11.11.0.160:8000/test1/index.html>

7. TestCase

- JUnitTest

7.1 Controller URL 히 테스트

```
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
import org.springframework.test.context.web.WebAppConfiguration;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;
import org.springframework.test.web.servlet.setup.MockMvcBuilders;
import org.springframework.web.context.WebApplicationContext;
import javax.inject.Inject;

@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = {"classpath:/spring/context-*.xml"})
@WebAppConfiguration
public class TestCase1 {
    @Inject
    private WebApplicationContext wac;

    private MockMvc mockMvc;

    @Before
    public void setUp() {
        this.mockMvc = MockMvcBuilders.webAppContextSetup(this.wac).build();
    }

    @Test
    public void myTest() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.get("/rba/selectUserList.do"));
    }
}
```

於

7.2 Service 호ft 테스트

```
import java.util.HashMap;
import java.util.Map;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
import org.springframework.test.context.web.WebAppConfiguration;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;
import org.springframework.test.web.servlet.setup.MockMvcBuilders;
import org.springframework.web.context.WebApplicationContext;

import com.southpeak.service.UserService;

@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = {"classpath:/spring/context-*.xml"})
@WebAppConfiguration
public class TestCase2 {
    @Autowired
    private UserService userService;

    @Test
    public void myTest() throws Exception {
        // 1. 사용자 정보 정의
        Map<String, Object> paramMap = new HashMap<String, Object>();

        // 서비스 호출
        userService.selectUserList(paramMap);
    }
}
```


7.3 DataSet (DataCollection, DataMap) 테스트

```

@Test
public void test1() throws Exception
{
    // DataMap 테스트
    // ( {"empId": "U00000001", "empNo": "dp.iturh"} )
    JSONObject jsonObj = new JSONObject();
    jsonObj.put("empId", "U00000001");
    jsonObj.put("empNo", "dp.klm");
    String jsonStr = jsonObj.toString();

    ObjectMapper mapper = new ObjectMapper();
    Emp emp = mapper.readValue(jsonStr, Emp.class);
    System.out.println("test1 : " + emp.getEmpId());
}

@Test
public void test2() throws Exception
{
    // DataCollection 테스트 (List<VO> 변환 테스트)
    // ( [{"empId": "U00000001", "empNo": "dp.kin"}, {"empId": "U00000002", "empNo": "dp.klm"}] )
    JSONArray jsonArray = new JSONArray();
    JSONObject jsonObj = new JSONObject();
    jsonObj.put("empId", "U00000001");
    jsonObj.put("empNo", "dp.kin");
    jsonArray.add(jsonObj);
    JSONObject jsonObj2 = new JSONObject();
    jsonObj2.put("empId", "U00000002");
    jsonObj2.put("empNo", "dp.klm");
    jsonArray.add(jsonObj2);
    String jsonStr = jsonArray.toString();

    ObjectMapper mapper = new ObjectMapper();
    Emp[] empList = mapper.readValue(jsonStr, Emp[].class);
    System.out.println("test2 : " + empList[0].getEmpId());
}

@Test
public void test3() throws Exception
{
    // DataCollection 테스트 (List<Map> 변환 테스트)
    // ( [{"empId": "U00000001", "empNo": "dp.kin"}, {"empId": "U00000002", "empNo": "dp.klm"}] )
    JSONObject jsonObj = new JSONObject();
    JSONArray jsonArray = new JSONArray();
    JSONObject jsonObj1 = new JSONObject();
    jsonObj1.put("empId", "U00000001");
    jsonObj1.put("empNo", "dp.kin");
    jsonArray.add(jsonObj1);
    JSONObject jsonObj2 = new JSONObject();
    jsonObj2.put("empId", "U00000002");
    jsonObj2.put("empNo", "dp.klm");
    jsonArray.add(jsonObj2);
    String jsonStr = jsonArray.toString();
    System.out.println(jsonStr);

    ObjectMapper mapper = new ObjectMapper();
    Object data = mapper.readValue(jsonStr, new TypeReference<List<Map<String, Object>>>() {});

    Map mapData = (Map<String, Object>) data;
    List listData = (List) mapData.get("userList");

    System.out.println("test3 : " + "Map" + listData.get(0).get("empId"));
}

```

7.4 DOM 테스트

```

import static org.junit.Assert.assertTrue;

import java.util.Map;
import java.util.Objects;

import org.apache.commons.beanutils.BeanUtils;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
import org.springframework.test.context.web.WebAppConfiguration;

// JAXP API
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

// Except _on
import org.xml.sax.ErrorHandler;
import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;

// W3C DOM Definition
import org.w3c.dom.Document;
import org.w3c.dom.DocumentType;
import org.w3c.dom.Entity;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;

import java.io.File;
import java.io.PrintWriter;

@RunWith(SpringJUnit4ClassRunner.class)
@WebAppConfiguration
public class DCHTest {
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Test
    public void test1() throws Exception {
        String fileName = "test.xml";
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
        DocumentBuilder db = dbf.newDocumentBuilder();
        Document doc = db.parse(new File(fileName));
        for(Node child = doc.getFirstChild(); child != null; child = child.getNextSibling()) {
            echo(child);
        }
    }

    private void echo(Node n) {
        int type = n.getNodeType();
        switch(type) {
            case Node.ATTRIBUTE_NODE:
                System.out.println("ATTR : ");
                printlnCommon(n);
                break;

            case Node.SECION_NODE:
                System.out.println("DATA: > ");
                printlnCommon(n);
                break;
        }
    }
}

```



```

case Node.CC , 예 NODE:
    System.out.println("COMM");
    printlnCo0Bo<i(n);
    break;

C*se Hode .DOCU , 田ft_JMGNEHT__m:
    System.out.println("DOC^FRAG:■");
    printlnCofnmon(n);
    break;

case Node. DOCUf^MTjmE:
    System.out.println("DOC :");
    printlnCoinRKx(n);
    break;

case Node.DOaxenniVfPEjmEz
    System.out.println("CELEH:");
    printlnCo__on (4);
    NaeedNodeMap nod eMap • ((DocuventType)n). getEntities();
    for(int i = 0; i < nodeMap. getLength(); i++)
    {
        Entity entity ■ (Entity)nodeMap.ite@i);
        echo(entity);
    }
    break;

case Node.£LEMCTnj_OD£:
    System.out.println("ELEM:");
    printlnCommon(n);
    NamedNodeMap atts ■ n.getAttributes();
    for(int i = 0; i < atts.getLength(); i++)
    {
        Node att ■ atts.item(i);
        echo(att);
    }
    break;

case Hode,ENTITY_NOOE:
    System.out.println("EMT:");
    printlnCofflmon( n);
    break;

case Node. ENTITY_REFEIKNCE_NOOE:
    System.out.println("ENTIREF:");
    printlnCommon(n);
    break;

case Node. NOTA TION_NOOE:
    System.out.println("NOTATION:M");
    printlnComncxi(n);
    break;

case Node. P9KKESSIN6_INSnUCTI(M_im£:
    System.out.println("PROC_INST:");
    println [o__on(n) ;
    break;

case Node.r£)rLMM>£:
    System.out.println("TEXT :");
    printlnCcMMon(n);
    break;

default:
    System.out.println("UNSUPPORTED NODE : * ♦ type);
    printlnCcMnon(n);
    break;
}
for(Node child ■ n.getFirstChild(); child != null; child ■ child.getNextSibling())
{
    echo(child);
}
}

```

```

private void printInCoraici(Node n)
{
    System.out.print(" nodeName=" + n.getNodeName() + " ");

    String val = n.getNamespaceURI();
    if(val != null)
    {
        System.out.print(" url=" + val + " ");
    }

    val = n.getPrefix();
    if(val != null)
    {
        System.out.print(" pre-X" + val + " ");
    }

    val = n.getLocalName();
    if(val != null)
    {
        System.out.print(" localName=" + val + "X");
    }

    val = n.getNodeValue();
    if(val != null)
    {
        System.out.print(" nodeValue=");
        if(val.trim().equals(""))
        {
            // Whitespace
            System.out.print("WS");
        }
        else
        {
            System.out.print(" " + n.getNodeValue() + " ");
        }
    }
    System.out.println();
}
}

```

7.5 SAX 테스트

```

import java.io.PrintStream;
import java.util.Enumeration;
import java.util.Hashtable;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

// JAXP API
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;

// SAX Parser
import org.xml.sax.Attributes;

// Exception
import org.xml.sax.ErrorHandler;
import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;
import org.xml.sax.XMLReader;
import org.xml.sax.helpers.DefaultHandler;

public class SAXTest extends DefaultHandler
{
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    private Hashtable t3gs^

```



```

public static void Min(String[] args) throws Exception
{
    String fileUrl = "file:///c:/DHRMISDi/workspace/DHRMIS/src/resources/spring/context*
dpp.xml";

    SAXParserFactory spf = SAXParserFactory.newInstance();
    spf.setNamespaceAware(true);
    SAXParser saxParser = spf.newSAXParser();

    SAXTest saxTest = new SAXTest();
    XMLReader xmlReader = saxParser.getXMLReader();
    xmlReader.setContentHandler(saxTest);
    xmlReader.parse(fileUrl);

    // System.out.println("..... * saxTest.tags);
}

public void startDocument() throws SAXException
{
    tags = new Hashtable();
}

public void startElement(String namespaceURI, String localName, String qName, Attributes attrs)
throws SAXException
{
    String key = localName;
    Object value = tags.get(key);

    if(value == null)
    {
        tags.put(key, new Integer(1));
    }
    else
    {
        int count = ((Integer)value).intValue();
        count++;
        tags.put(key, new Integer(count));
    }

    System.out.println(key);
    int attrCount = attrs.getLength();
    for(int i = 0; i < attrCount; i++)
    {
        String qName1 = attrs.getQName(i);
        String value1 = attrs.getValue(i);
        // String url = attrs.getURI(i);
        // String localName = attrs.getLocalName(i);
        System.out.println(" * qName1 * " + value1);
    }
}

public void endDocument() throws SAXException
{
    Enumeration e = tags.keys();
    while(e.hasMoreElements())
    {
        String tag = (String)e.nextElement();
        int count = ((Integer)tags.get(tag)).intValue();
    }
}

private class MyErrorHandler implements ErrorHandler
{
    private PrintStream out;
    MyErrorHandler(PrintStream out)
    {
        this.out = out;
    }
    private String getParseExceptionInfo(SAXParseException spe)
    {
        String systemId = spe.getSystemId();

```

```

        if(systemId == null)
        {
            systeold « "null";
        }
        String info = "URI«" + systemId + "** Line«" + spe.getLineNuaber() + "-j" +
        spe.getMessage();
        return Info;
    }

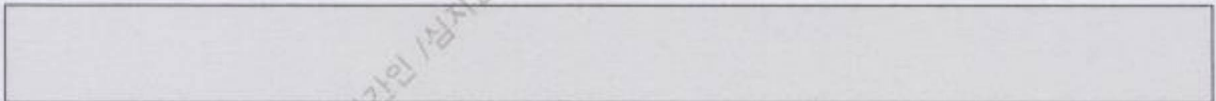
    @Override
    public void warning(SAXParseException spe) throws SAXException
    {
        System.out.println("Warning : " + getParseExceptionInfo(spe));
    }

    @Override
    public void error(SAXParseException spe) throws SAXException
    {
        String message = "Error : " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }

    @Override
    public void fatalError(SAXParseException spe) throws SAXException
    {
        String message = "Fatal Error : " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }
}
}
}

```

7.6 POI 테스트



8. 會투션

8.1 OZ Report

가. 레포트 생폴

0미 샘플

```
<body ev:onpageload«"scwin.onpageLoad" ev:onpageunload■轉scwin.onpageun:<xx/>»
  <div id«*testl* style*iridth;753px;height:791px; *></div>
</body>
```

```
scwin. or)ffMgelo«d • fvncion()
```

```
<
```

```
ver opt • ();
```

```
opt. filcnaM ■ 'B6J>5,( » , a ) , 206,687B1038,81, 23, SO, 47, 9F, EF,E2,Q3,75,C9,55,38(68,) .EC,23';
```

```
opt.uri ■ "/sys/exp/tst/s«lectJsonTest2.do{
```

```
opt.p*raal ■ *13,2, FA,X , M,M,9A,M,A2,的, 的, E2,U,*3«90,2A';
```

```
coa.report.odiFora('test\ opt);
```

```
};
```

submit

D 11.11.01.160.2003-cc9/odform.jsp - Chrom : C- . *

① 주의 ft蓄 11.11.요 1 的:,0的/oy的/od나nrm. jj分

을 往! 函

AUTH0AX00E	AUTHOR_NM	AUTHOR JK	AUTHORJCREATDE
ROCE00000I	슈 B휴 지	天上天下 鳴稀簡«이13-08-30	
ROLE000002	시스템 담당자	mil data 2011-1-08-07	
AOLE000003	■인자繫	o o	20/09/09
AOLE000004	보직신성	O D	2(yoy的
ROCE0000Q5	»원업 무	ndl data	2OW的
ROCE0000D6	지 ««친	mil dale	2(yoa<的
ROLE0000D7	진급업무	ndl data	2QW09
ROLE000008	정업무	ndl d«i寒	2(yoyo9
AOLE00000的	4계관리	ndl data	2 이oyo9

문비 이시지 1 / 1

JSON 샘플

```
<body ev:onpageload*"scwin.onpogetoad" ev:onpageunload*"scirin.onpageunLoad">
  <liv id="testl" style="Width:753px;height:TB1px;"></div>
</body>
```

sctrin. onpageload • fuactionO

```
<
  var opt • {} ;
  opt.fllw^e -
  opt.url ■ '/sys/cxp/tst/MI«ctJsofiTcst4.do*';
  opt. par ml -
  COB.report.js«nFor»( 'testl. • opt);
};
```

22<E> >aner.)»t

MW11

submit

B > 1 10 - ? < 01 : 00 , " , _ , , [■ ""n

必 주의 요항 11JK0J6O_>c .az幻分wnfomwsp



[|그“



100%

@ is! Si

regtHepon2.ozr

atchFileid	creatDt	utaYn
mil ctota	的的.0&I9 10:43: 的	V
nJI data	2C20.0&I9 10:52:27	V
mil data	2020.06.19 10:54:30	Y
n_ “	2020.06.19 IO:4(M9	Y
mil data	2020.06.19 10내公 09	Y
mil data	2020.06.24 17:17:25	Y
ndi data	2C20.06.31 16:24:29	Y
ndI data	2020.08.31 16 : 26 : 08	Y
nUI data	2020.06.31 16:27:23	Y
mil data	202a(»3I l&2&5I	Y

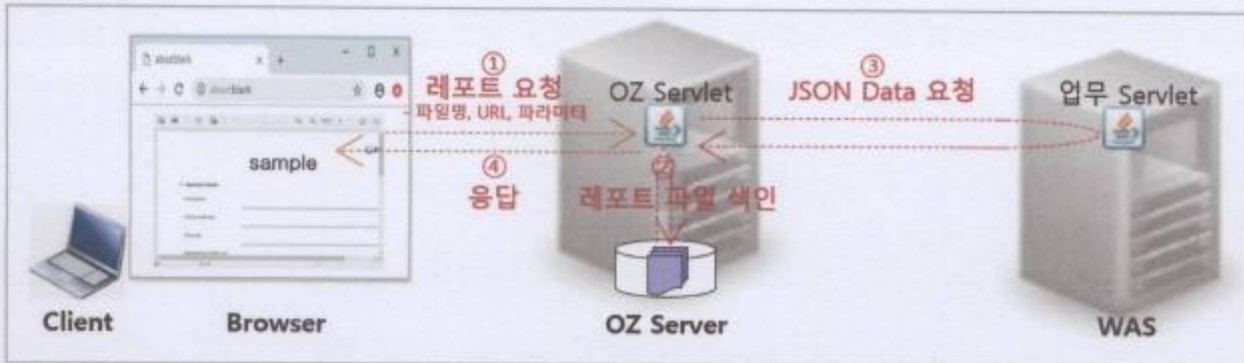
한도

■너지

1 / 1

나. Client Report 생성 요청

- 개념도



(1) HTML5 Canvas Viewer

- HTML5 <canvas> 이용해서 Report 생성
- 서버에서 서식파일과 Data 을 따로 내려주어 Client 에서 조립하여 본다.
- Client가 적당한 부하를 감수하고자 할 때 사용.
- Data 10 ~ 50 건 사이 정도 의 데이터 처리시에 사용 권장.

• 예제

```

<script type="text/javascript" src=7oz80/ozhviewer/OZJSViewer.js" charset="utf-8"> </script>

<script type="textZjavascriptM" >
    var jsonStr = .fTestData::[f.key":_ar, -name": "br", rke/_a2", 'nameVb2"}J].;

    function SetOZParameters_OZViewer() {
        var oz = document.getElementById(OZViewerH);
        oz.sendToActionScriptCconnection^ervletw, httpy/11.11.0.160:8000/oz80/serverM);
        oz.sendToActionScript(connection.reportname^, 7sample.ozr); OZ Report 파일 위 대
        oz.sendToActionScript(Mtoolbar.save.*_HfalseM);
        o내ndToA_nScriptrtoolbar.p_nnt*.그^^ e"); → 인쇄 미리보기 화면에서 저장/인쇄 버은 조정
        oz3endToActionScriptrexpport.applyformat\ 'xls,xlsx ', doc"); → 미리보기에서 저장가능 타입 지:

        return true;
    }

    startjD 리 s("OZViewer", 7oz8(yozhviewer/**) ;
</script>

```

(2) HTML5 SVG Viewer

- HTML5 <svg> 이용해서 Report 생성.
- 서버에서 서식파일과 Data 을 조립하여 Client 에 내려준다.
- S0 건 이상의 데이터 처리시에 사용 권장.
- 예제 八災O^zhviewe^OZJSSVGViewer.js

```
<script type="text/javascript" src="7oz80/ozhviewer/OZJSSVGViewer.js" charset="utf-8">
```

```
<script type="text/javascript">
```

수 SVG Viewer Saipr 선언

```
var jsonStr = TTestData: {rkey:"ar, _name":"brikey":"a2\ "name":"b2w}}*};
```

```
function SetOZParameters.OZViewerO {
```

```
    var oz = document.getElementById("OZViewer");
```

```
    oz.sendToActionScriptConnection.srvlet" ; 'http://11.11.0.160:8000/oz80/server';
```

```
    oz.sendToActionScriptConnection.reportnameT/sample.ozr"); -> OZ Report 파일 위
```

```
    oz.sendToActionScriptCtoolbar.save\ **false");
```

```
    oz.sendToActionScriptCtoolbar.print\ **false"); -> 인쇄 미리보기 화면에서 저장/인쇄 버튼 조정
```

```
    oz.sendToActionScriptCexport.applyformat\ **xls,xlsx,doc"); -> 미리보기에서 저장가능
```

타입 지:

```
    return true;
```

```
}
```

```
var opt = {};
```

```
opt.rendering_mode**] = "svg"; -> SVG로 처리하기 위한 옵션
```

```
start_o진 s(_OZViewer" , **화 z80八)zhviewer";
```

```
</script>
```

(나,

(3) Exe Viewer

- Exe CS Program 을 이용

善

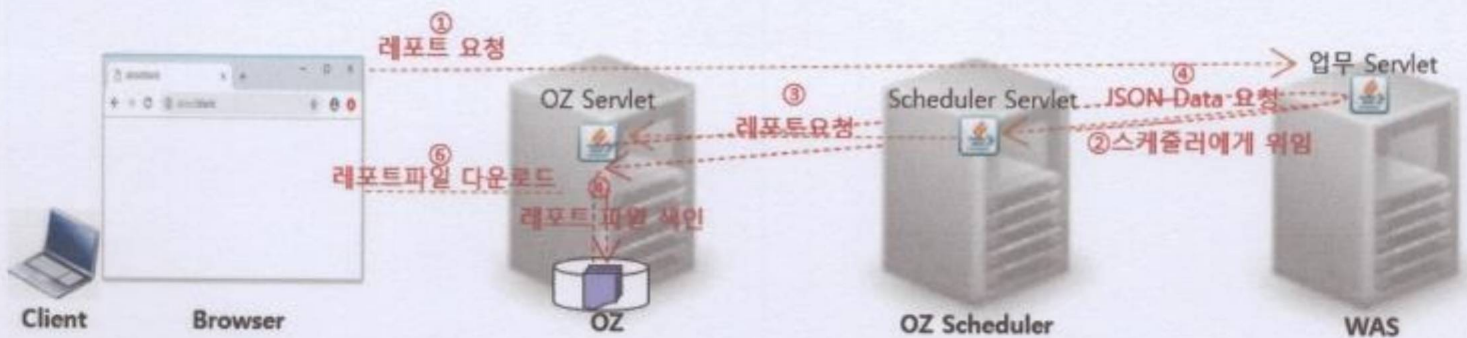
X 표준버전으로 변경되면서 접근성 측면에선 좋아졌으나, 속도면에서는 더 빨라졌다.

배

버전이 업그레이드 되었다고 속도가 더 빨라졌을거라는 오해 하지마세요.

다. Server Report 생성 요청

- 개념도



- 예제

```
String ozServerUrl ■ "http://11.11.0.160:8002/oz80";
String ozAdmin = admin;
String ozPasswd ■ "password";

Serverinfo serverinfo = new ServerInfo();
serverinfo.setID(ozAuthId);
serverinfo.setPWD(ozAuthPw);
serverinfo.setIsDaemon(false);
serverinfo.setURL(ozServerUrl);

SortProperties configMap = new SortProperties();
configMap.setProperty("task_type", "viewerTag");
configMap.setProperty("launch_type", "immediately");
configMap.setProperty("cfg.type", "new");

SortProperties exportMap = new SortProperties();
exportMap.setProperty("connection.servlet", ozServerUrl);
exportMap.setProperty("connection.reportname", "overview.ozr");
exportMap.setProperty("export.format", "pdf");
exportMap.setProperty("pdf.filename", "test-report");

exportMap.setProperty("child1.connection.servlet", ozServerUrl);
exportMap.setProperty("child1.connection.reportname", "testreport1.ozr");
exportMap.setProperty("child1.export.format", "pdf");
exportMap.setProperty("child1.pdf.filename", "test-child-report.pdf");

exportMap.setProperty("child2.connection.servlet", ozServerUrl);
exportMap.setProperty("child2.connection.reportname", "testreport2.ozr");
exportMap.setProperty("child2.export.format", "pdf");
exportMap.setProperty("child2.pdf.filename", "test-child-report.pdf");

ozScheduler.directExport(serverinfo, configMap, exportMap);
```

來 하나의 PDF 로 Merge 하고자 할때는 `exportMap.setProperty(global.concatpage*, "true")` 와 같이 작성.
 技 개별 PDF 로 나눠 출력하고자 하면 `exportMap.setProperty ("global.concatpage", _false` 가로 작성한다.
 來 각 레포트에서 데이터가 많을 경우 concurrent 룰 주고, pageque 쿨 적정하게 나눠준다.

X 데이터톨 요청하는 방법으로 여러가지가 있는데, 크게 업무서비스 URL을 호출하거나, 직접 쿼리물 호출하거나 하는 2가지 방법이 있다. (다음장에서 논의.)

X JSON vs ODI

1.JSON 방식

- 업무 URL 호출.

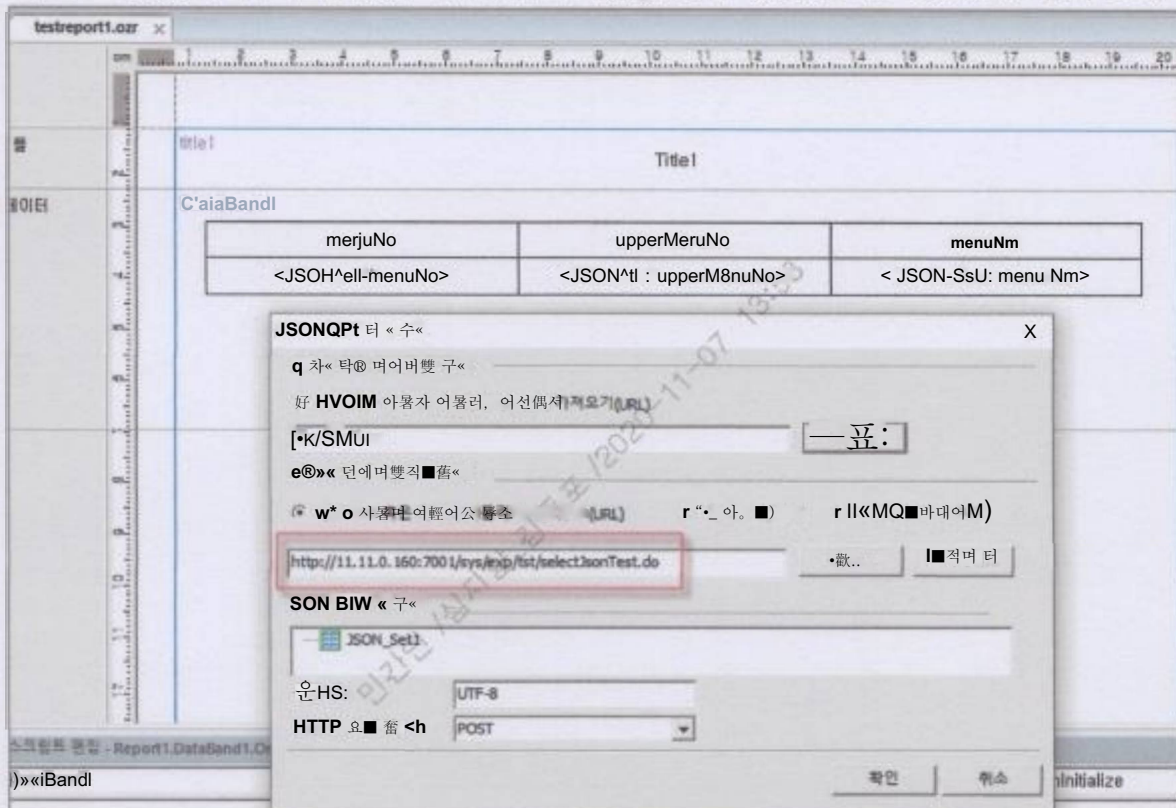
- 장점

업무쿼리, 레포트 쿼리 2벌 작성 필요 없음.

레포트가 유출될시 보안에 위험.

- 단점

레포트 서블릿이 과부하 걸릴경우 업무WAS가 Hang걸릴수 있음. 별도의 Container 분리 고려,



2. ODI 방식

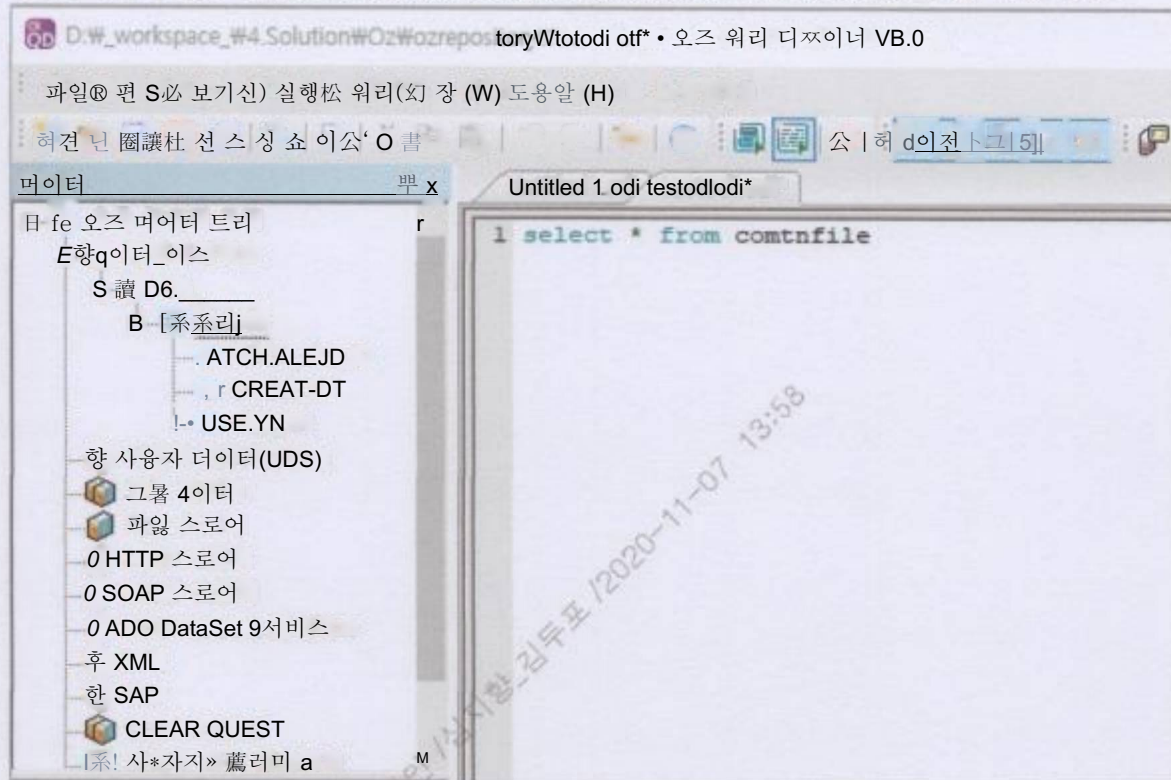
-OZR 파일에 SQL 작성.

- 장정 :

ASIS 소스를 이용할수 있다함.

- 단정

OZR 파일 호출시 odi.odinames, odi.testodiodinames.pcount, odi.tesodi.args 등 작성 필요



於

8.2 MarkAny

-OZ 는 Report 에 Water Mark 인쇄하는 기능

- OZ Report 호출하는 스크립트에 Option 추가하는 형태로 연동됨.

```

<script type="text/javascript" src="7oz80/ozhviewer/OZJSSVGViewer.js" charset="utf-8"> </script>
<script type="text/javascript" >
    var param = TsearchTitleTtest0)*;

    function SetOZParamters.OZViewerO {
        var oz;
        oz = document.getElementByIdC'OZViewer);
        oz.sendToActionScript("connection5ervlefrhttp/11.11 义).160:8000/oz80/server");
        oz.sendToActionScriptC · connection.reportnameH//sample.ozrM);
        oz.sendToActionScript ("toolbar^ave' "false");
        oz.sendToActionScript("toolbar.print", "false");
        oz.sendToActionScript("export.applyformat*", Mxls_xlsx#docH);

        oz.sendToActionScript('print.externalmodule\'oz.viewer.export.02^AppletPrintBarcode_MarkAny');
        oz.sendToActionScriptCconnection.extraparam·7 · PropertiesFilePath=/u01/jeus/webapps/oz80/WEB-
INF/cohf/markany.properties^UserPasswordsO^dfCreatorsMarkAny*);

        return true;
    }

    var opt = 0;
    optr'print_exportfromH] = "scheduler*";
    start_ozjsCOZViewer"R/oz80/ozhviewer/* # opt);
</script>

```

MarkAny 처리

X 레포트 뷰어에서는 바코드가 안나오고, 인쇄버튼을 눌러 미리보기시 바코드가 출력됨.

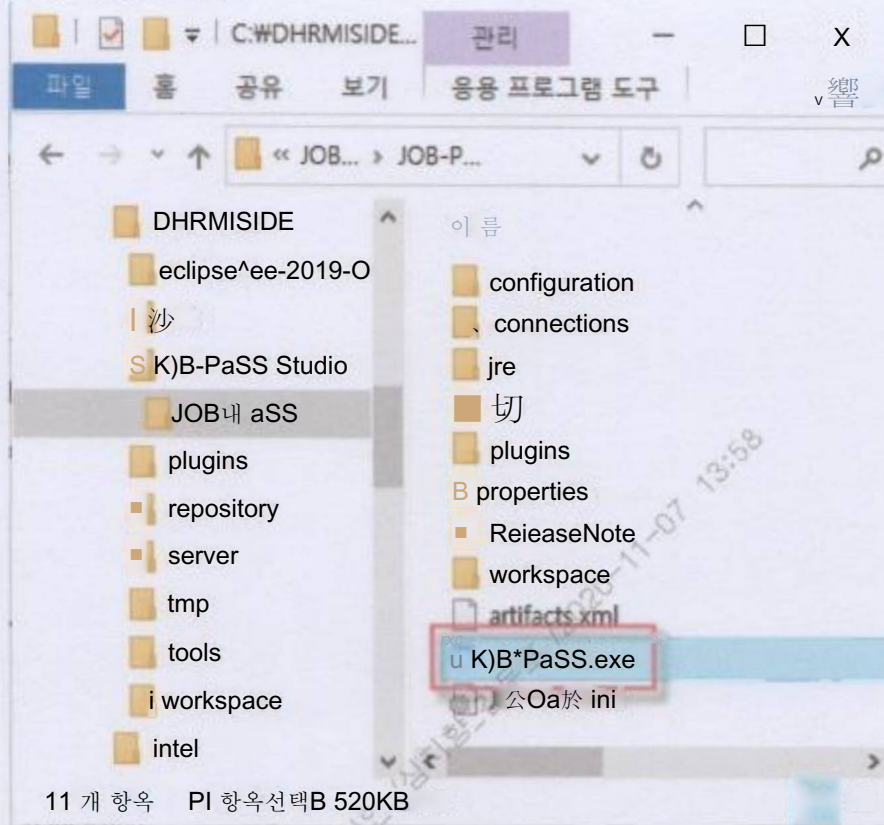
솔루션 업체에서 제공하는 바코드스캔 업을 통해 위변조된 레포트인지 원본인지 확인 가능.

8.3 JOB-PaSS

- Eclipse Framework 기반 솔루션

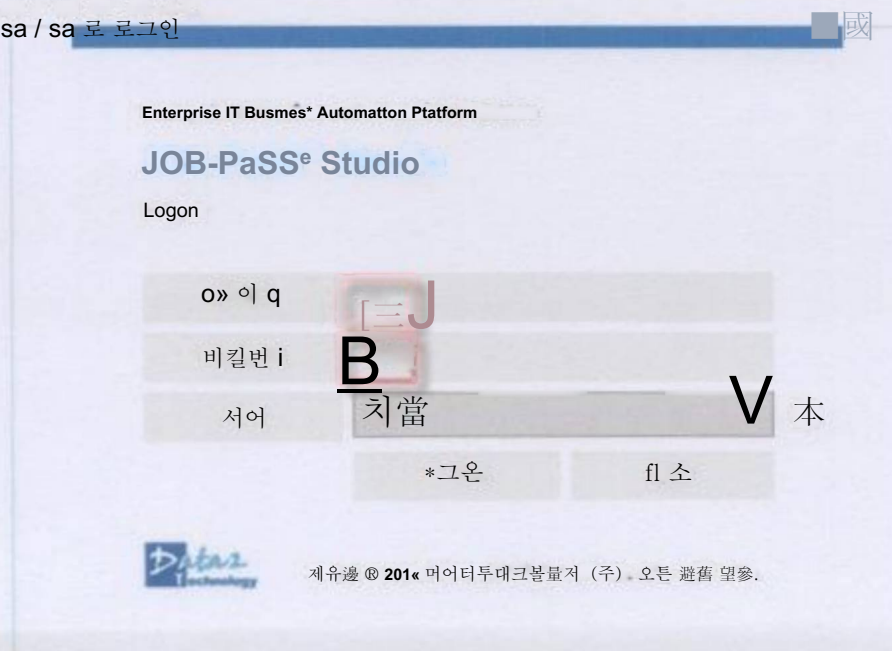
가. 실행

JOB-PsSS.exe 실행



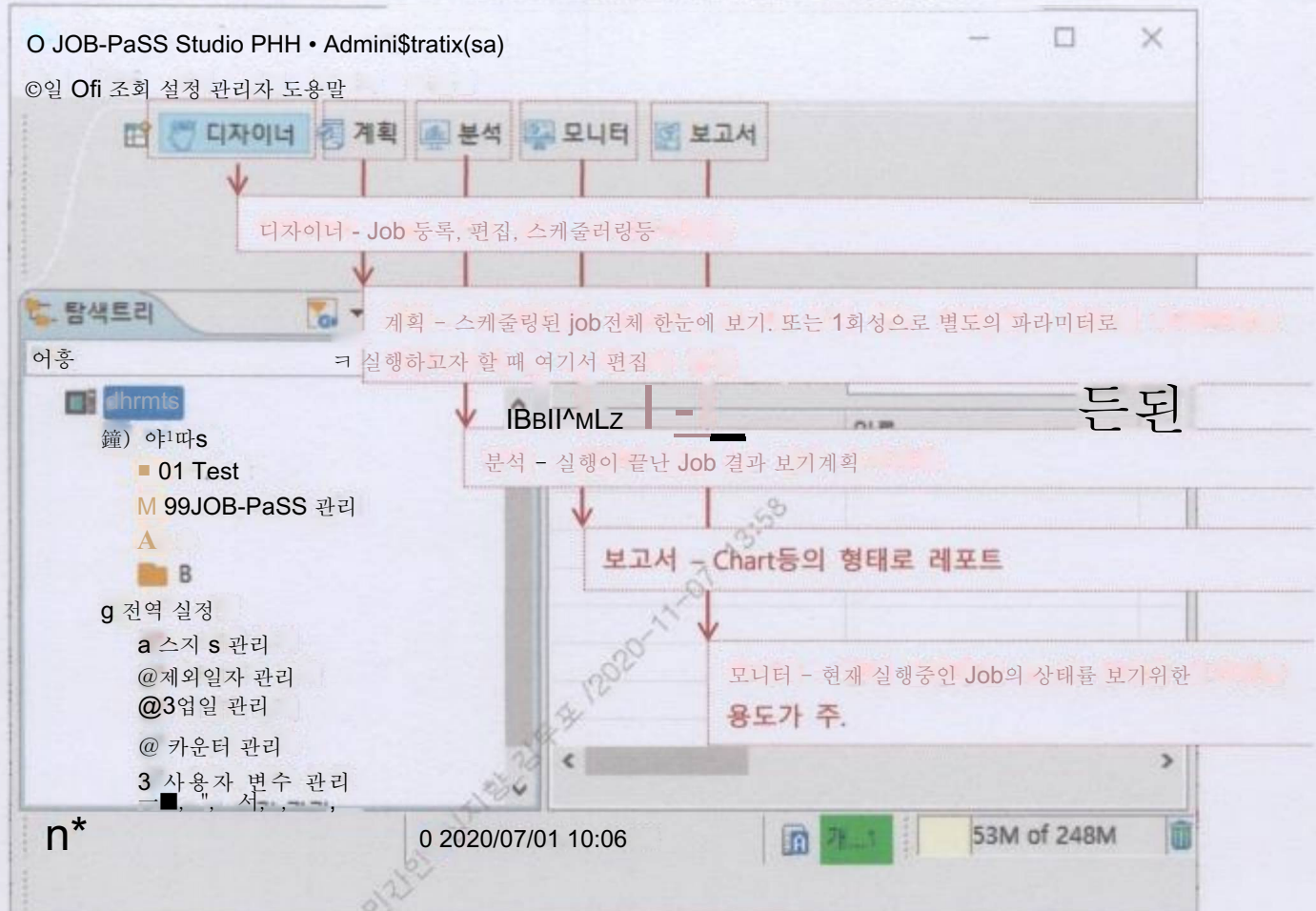
나. 로그인

sa / sa 로 로그인



다. 개요

- **Perspective**는 총 가지로 디자이너, 계획, 분석, 모니터, 분석이 있고 이 중 설계자나 개발자■은 주로 디자이너를 사용하게 됨.



M 이 외에 관리자 기능에

접속정보 설정 - Scheduler Agent 등록/편집/삭제

서버설정 시ob들이 실행후 남기는 로그들 삭제 Rolling 주기 설정등

사용자권한 설정 - 해당 로그인 사용자가 사용할수 있는 컴포넌트 종류 설정등 (Job 종류)

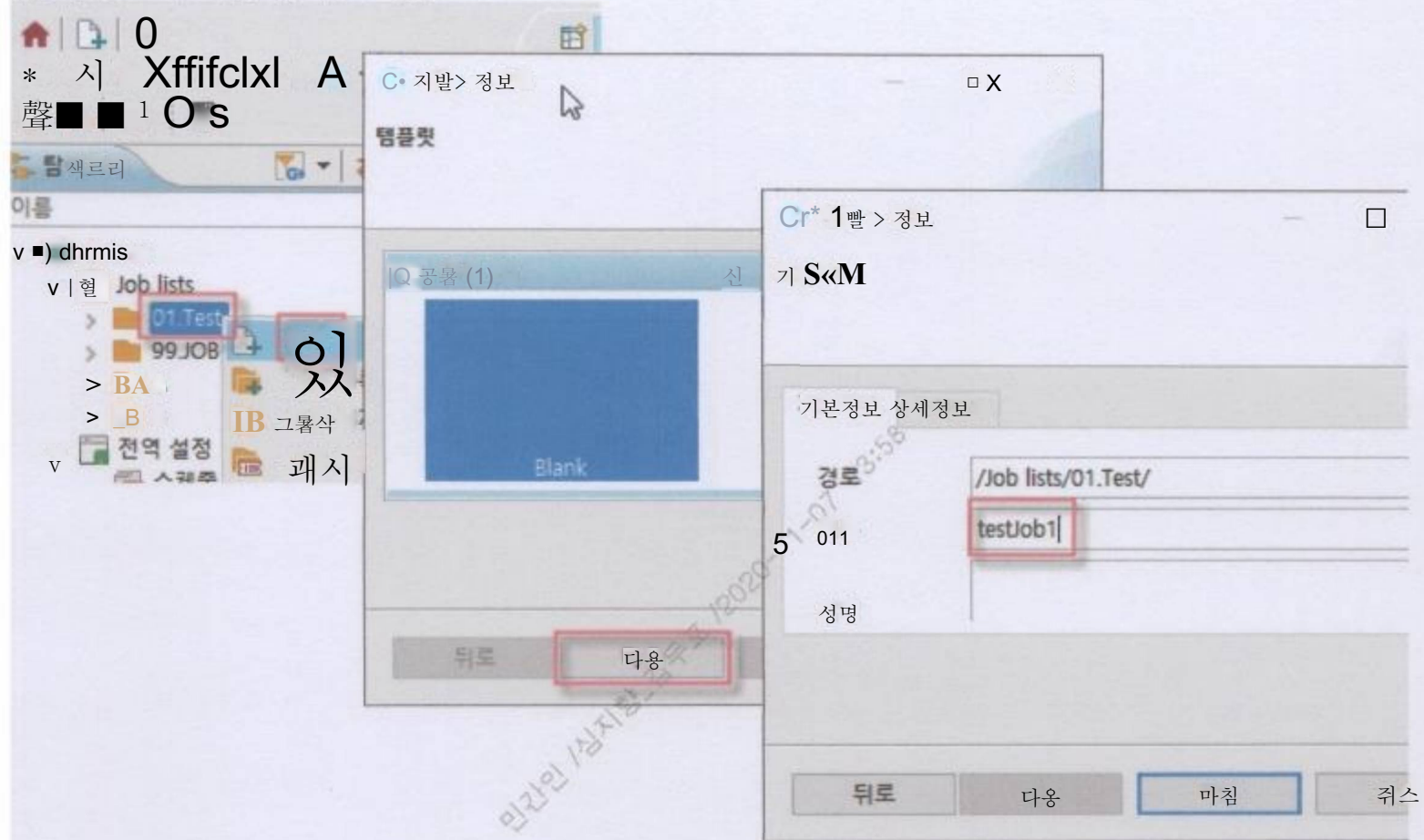
등의 기능이 있으나 공통단에서 처리할 예정임.

라. Job 생성

-jobjists 선택 > 해당 업무그룹 선택 > 등록

G. K>B-PaSS Studio [Tilt] - Administrator(sa)

머« 편집 조회 설정 관리자 도움말

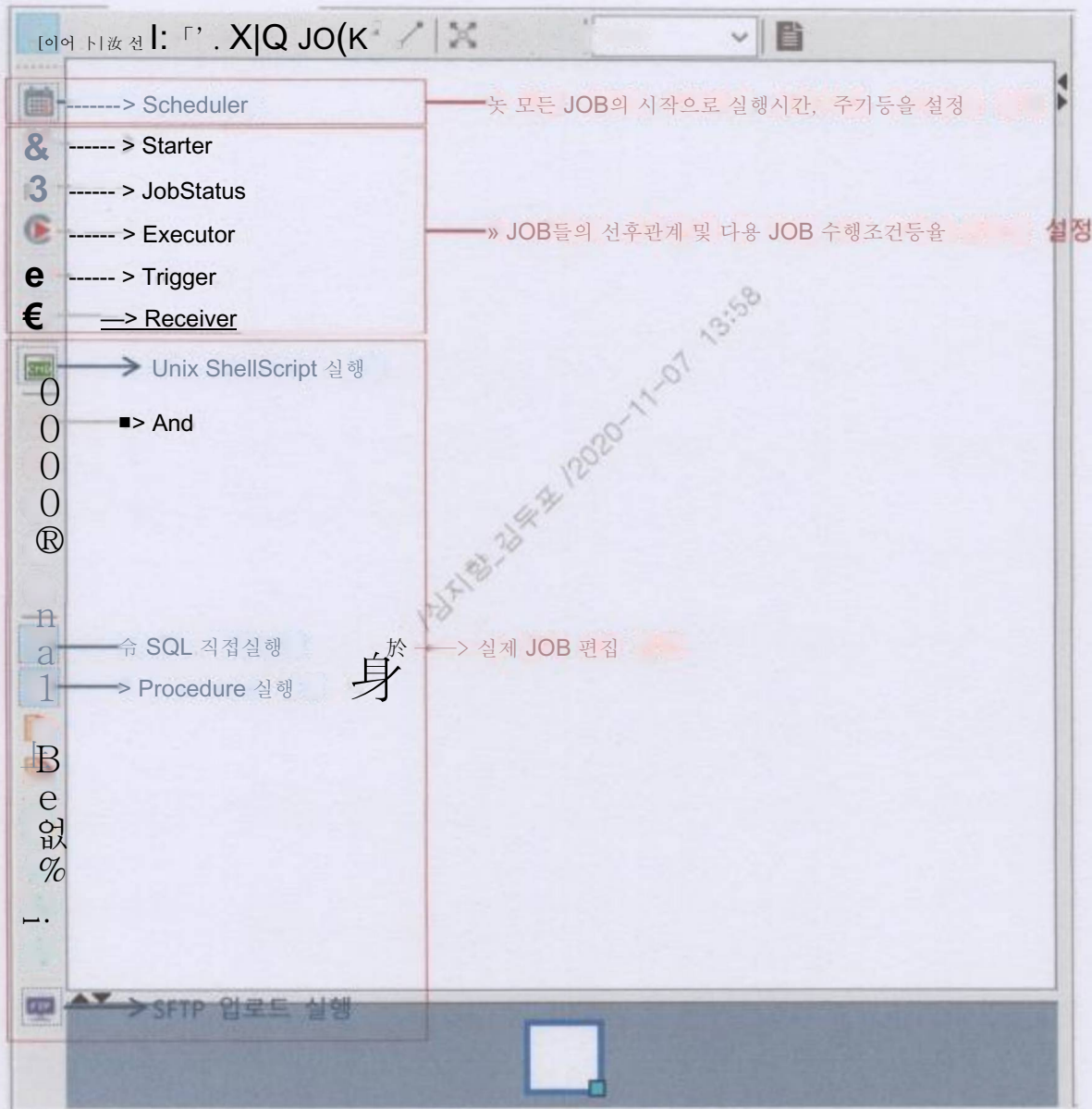


마Job 편집

1. Job 편집시 시 4는 Scheduler, Starter, Receiver 셋 중 하나인데, 대부분 Scheduler에서 시작.
2. Job의 종류로는 SQL, StoredProcedure, Unix ShellScript SFTP 등이 있다.
3. Starter, JobStatus, Executor, Trigger, Receiver, And는 각 Job들의 선후관계 및 조건등을 걸고자 사용하는 기능으로 필요시 Executor* 사용하기를 권장.

采 Version Up History : Trigger/Receiver > JobStatus > Excutor

※



바. 예제

(1) Unix ShellScript 등록

- 스케줄러 등록 > Run Program BI 등록 > 링크 연결

> JOB*^«SS Sbxfto ptV; • AdmtrmtrM0r(MI)

IB 9 끝 조繼I 傷정 조작자 도물알

錢卜YB

*시xfflfcixia , 分

B r q 자어너 쏘 저鷄 *식 . 요니더

할색트러 ■»■, 다, 7 0미 Qd 색:L公, HUOb, a

가를 뉴 今 “신.”, :: ::|Q| , - B

!■ dhrmh

Job lists
testJob1
99XJe-P«SS W
A

oeTOrw 서버로그 정리 작업
RunProorani

<> 1番, 선, 다어雷嚴그

에어전르 W0M9■口

S 置 IAJOVK必P«M혀 oagent/workspace/

과S3 twtiofoh

여 • 크: 권한 수S 시간

test-log-clear.sh 7 rwx 2020/07/01 13:40:3

확인

分* 7■舊 > test-log-clear - RunProgrvn

기본 «*

상세 정보

고 급 호 «

요휴관리

기타

O■어 5«

대표£그8

서작 調지

확라머 터

이■M.AgncC

test-log-ctev ih

AiOI^obpasMobdgem/mxkspdce/

사-1

파라미터
1 333

+ 추가 _삭禪 ♦ 위& ♦ 아래1

« 회정

■ »물

사물자

먹그화표드 舊,

ecumtSMc V

선9작* »量 專 ap|시간 :

SIQ 실관 서갓

머물

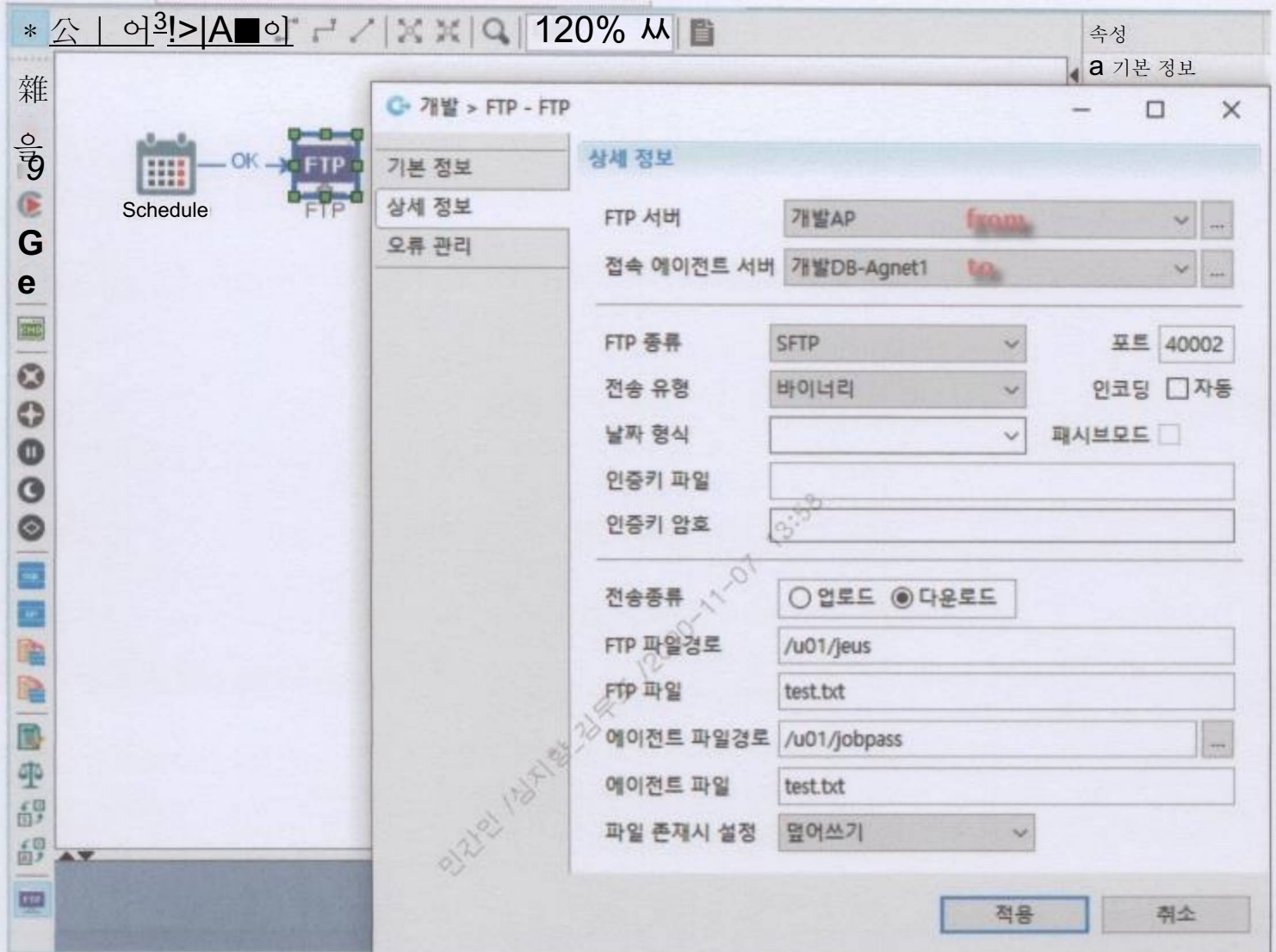
■土

© 2020AJ7/01 1332

개발 | 11.11.2023 | 1451

(4) SFTP 실행

- 스케줄러 **M** 등록 >, 등록 > 링크 연결



V-----

(5) 조건 설정

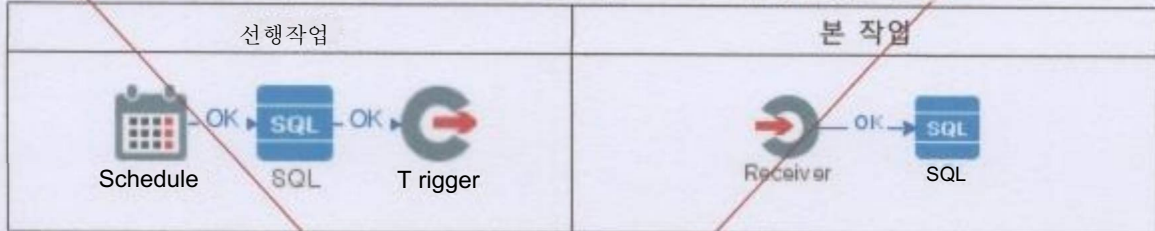
-2 개이상 Job 전후관계, 조건등 설정

1. Trigger/ Receiver 이용

- Trigger 와 Receiver 가 Set 로 동작한다,

- 역행 작업은 끝에 G Trigger 로 끝을 내고, 본작업은 S Receiver 로 시작:

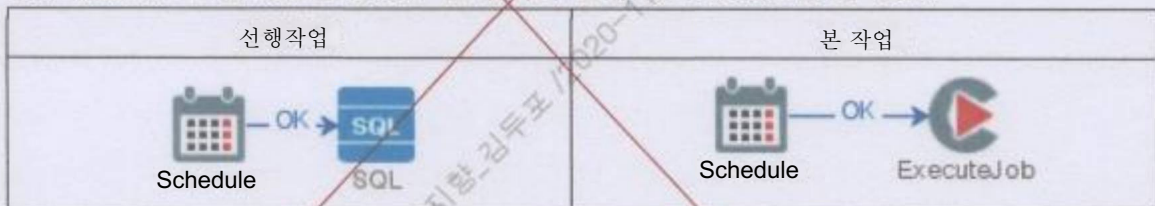
• 1\1 로만 작동하고, 사실상 Deprecaed 된 기능으로 쓰지 않을것을 권장/



2. Executor 사용

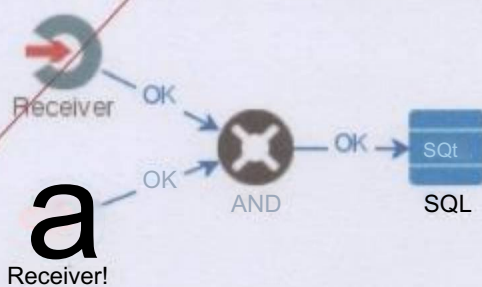
- Executor 와 Starter 가 Set 동척한다.

• 현재 Job 의 Flow 중 한 노드에 [>트 Job 을 호록하고자 할 때 Executor € 톨 사용하는데, 실제 Executor 가 될 Job 을 생성시 앞 에 딱히 둘게 없어 Starter를 둔 형태.



3. AND 이용 /스 <公

• 2 가지 이상_업이 끝나야 본작업이 실행되어야 할 경우 AND



V

4. JobStatus 사용

- AND O 보다 업그레이드 된 형태로 JobStatus 3 안에 선행되어야 할 Job 목록을 등록해서 사용.



Schedule JobStatus SQL

개발 > JobStatus - JobStatus

기본 정보

상세 정보

오류 관리

상세 정보

선행 작업 목록

종료 시간

경로	이름	선행 ExecuteJob
/Job lists/01.Test/	precedingJob1	
/Job lists/01.Test/	precedingJob2	

+ 추가 - 삭제 * 등록된 작업은 모두 AND 연산적용

결과 조건 ☒ 모든 조건 충족시까지 대기

대기 구분 무한 대기 v

적용

취소

旋

9. 시큐어 코딩

9.1 전자정부 Code Inspection RuleSet 39 개

No.	Rule 이름	비고
1	EmptyCatchBlock	
2	EmptyIfStmt	
3	EmptyWhileStmt	
4	EmptyTryBlock	
5	EmptyFinallyBlock	
6	UnnecessaryConversionTemporary	
7	EmptyStatementNotInLoop	
8	WhileLoopsMustUseBraces	
9	AssignmentInOperand	
10	UnnecessaryParentheses	
11	SimplifyBooleanExpressions	
12	SwitchStmtsShouldHaveDefault	
13	AvoidReassigningParameters	
14	FinalFieldCouldBeStatic	
15	EqualsNull	
16	SimpleDateFormatNeedsLocale	
17	ImmutableField	
18	AssignmentToNonFinalStatic	
19	AvoidSynchronizedAtMethodLevel	
20	AbstractClassWithoutAbstractMethod	
21	UncommentedEmptyMethod	
22	AvoidConstantsInterface	
23	DuplicateImports	
24	ImportFromSamePackage	
25	SystemPrintln	
26	VariableNamingConventions	
27	MisleadingVariableName	
28	AvoidArrayLoops	
29	UnnecessaryWrapperObjectCreation	
30	AvoidThrowingRawExceptionTypes	
31	AvoidThrowingNullPointerException	
32	StringInstantiation	
33	StringToString	

34	InefficientStringBuffering	
35	InefficientEmptyStringCheck	
36	UselessStringValueOf	
37	UnusedPrivateField	
38	UnusedPrivateMethod	
39	UnusedFormalParameter	

9.2 일반 코드 품질개선 예제

Rule-1	EmptyCatchBlock
설명	내용이 없는 Catch Block 존재. Exception 처리가 안될 수 있음.
조치방법	catch시의 처리내용을 추가할것
잘못된 예제	<pre> public void doSomethingO { try { FileInputStream fis = new FileInputStream(7tmp/bugger"); } catch (IOException ioe) (») z </pre>
올바른 매제	<pre> public void doSomethingO { try { FileInputStream fis = new FileInputStream(7tmp/bugger"); } catch (IOException ioe) (doSomethingO ; } } </pre>
Rule<2	EmptyFinallyBlock
설명	finally block이 비어있음
조치방법	빈 블록 삭제 < finally 내부 로직 추가>
잘못된 예제	<pre> public class Foo { public void barO { try { int x=2; } finally (// empty! } } } </pre>

올바른 예제	<pre> public class Foo { public void barO (try { int x=2; } finally { int y = x+3; return y; } } } </pre>
Rule-3	EmptyIfStmt
설명	빈 if 구문의 사용을 피하도록 함
조치방법	삭제 또는 반대 조건을 이용하여 처리
잘못된 예제	<pre> public class Foo { void bar(int x) (if (x == 이 { // empty! }) } </pre>
올바른 예제	<pre> public class Foo (void bar(int x) (if (x == 0) { doSomething(); } else if { doSomethingOtherf); } } } </pre>
Rule-4	EmptyStatementNotInLoop
설명	필요없는 문장 (V) 이 있음
조치방법	삭제
잘못된 예제	<pre> public class MyClass { public void doItO { // the extra semicolon here this is not necessary System.out.println(Clook at the extra semicolon**);; } } </pre>
올바른 예제	<pre> public class MyClass { </pre>

	<pre> public void doitO { // the extra semicolon here this is not necessary System.outprintlnriook at the extra semicolon"); } </pre>
Rule<5	EmptyTryBlock
설명	내용이 없는 try 블록이 존재함
조치방법	삭제
잘못된 예제	<pre> public class Foo (public void barO { try { } catch (Exception e) { e.printStackTraceO; } } } </pre>
올바른 예제	<pre> public class Foo (public void barO { } } </pre>
Rule-6	EmptyWhileStmt
설명	빈 while 구문이 사용되었음
조치방법	삭제
잘못된 예제	<pre> public class Foo { void bar(int a, int b) { while (a == b) (//) } } </pre>
올바른 예제	<pre> public class Foo { void bar(int a, int b) (//소스 구현 없는 경우 메소스, 콜래스트도 삭제) } </pre>
Rule-7	UnnecessaryConversionTemporary
설명	기본 데이터 (primitive type) 를 String으로 변환할 때 불필요한 임시 변환 작업을

	피하도독 함
조치방법	각 primitive type 을 대표하는 클래스 ■ <Integer<character , Double...^ static method를 이용하여 변환처리
»못된 예제	<pre> public String convert(int x) { // this wastes an object String foo = new Integer(x).toString(); } </pre>
올바른 예제	<pre> public String convert(int x) { return Integer.toString(x); } </pre>
Rule-8	WhileLoopsMustUseBraces
설명	중괄호 ({ }) 없이 사용된 while문의 사용은 피하라
조치방법	중괄호 추가
잘못된 예제	<pre> public void doSomethingO { while (true) x++; } </pre>
올바른 예제	<pre> public void doSomethingO { while (true) { x++; } } </pre>

9.3 코드 가독성 개선 예제

Rule-9	AssignmentInOperand
설명	피연산자내에 할당문이 사용됨. Code 를 복잡하고 유지보수성이 떨어지게 만듦
조치방법	유지보수성을 위해 각각을 나누어 처리할것
잘못된 예제	<pre> public class Foo { public void barO { int x = 2; if ((x = getXO) == 3) { doSomethingO ; } } private int getXQ { </pre>

	<pre> return 3; } } </pre>
올바른 예제	<pre> public class Foo { public void bar() { int x = 2; x = getXO ; if (x == 3) { doSomethingO ; } } private int getXO { return 3; } } </pre>
Rule-10	Un necessaryParentheses
설명	괄호가 없어도 되는 상황에서 불필요한 괄호를 사용할 경우 마치 매소드 호출처럼 보여서 소스 코드의 유지보수성을 떨어뜨릴 수 있음. 이에 불필요한 괄호를 삭제토록 함
조치방법	괄호삭제
잘못된 예제	<pre> public class Foo (boolean barO { return (true); } 승 } </pre>
올바른 예제	<pre> public class Foo (boolean barO { return true; } } </pre>

9.4 코드 설계 변경으로 품질개선 예제

Rule-11	SimplifyBooleanExpressions
설명	boolean 사용 시 불필요한 비교 연산을 피하도록 함. 단순한 code 를 복잡하게 만듦.
조치방법	불필요한 비교연산 대신 단순연산처리

잘못된 예제	<pre> public class Bar { private boolean bar = (isFooO == true); public isFooO {return false;} } </pre>
올바른 예제	<pre> public class Bar { bar = isFooO ; public isFooO {return false;} } </pre>
Rule-12	SwitchStmtsShouldHaveDefault
설명	Switch구문에는 반드시 default label이 있어야 함
조치방법	default label 추가
잘못된 예제	<pre> public class Foo (public void barO { int x = 2; switch (x) (case 2: int j = 8;) } } </pre>
올바른 예제	<pre> public class Foo { public void barO { int x = 2; switch (x) { case 2: int j = 8; default doSomethingO ; } } } </pre>
Rule-13	AvoidReassigningParameters
설명	할당을 통해 파라미터 값을 직접 변경금지
조치방법	필요하다면 임시지역변수를 사용함.
잘못된 예제	<pre> private void foo(String bar) { bar = "something else*"; } </pre>
올바른 예제	<pre> private void bar(String bar) { String tmp = 'something else**'; } </pre>
Rule-14	FinalFieldCouldBeStatic

설명	final field# Static 으로 변경하면 overhead 를 줄일 수 있음.
조치방법	static 추가
잘못된 예제	<pre> public class Foo { // this could be static and save some space public final int BAR = 42; } </pre>
올바른 예제	<pre> public class Foo { // this could be static and save some space public final static int BAR = 42; } </pre>
Rule-1S	EqualsNull
설명	「null 값과 비교하기 위해 equals 함수를 사용하였음.
조치방법	null 비교는 (x == null) 형태로 변경
잘못된 예제	<pre> class Bar (void fooO { String x = "too"; if (x.equals(null)) { doSomethingO ; } }) </pre>
올바른 예제	<pre> class Bar { void foo() { String x = "foo"; if (x == null) { doSomethingO ; } } } </pre>
Rule-16	SimpleDateFormatNeedsLocale
설명	SimpleDateFormat 인스턴스를 생성할때 Locale 을 지정하는 것이 바람직함
조치방법	Locale 명시
잘못된 예제	<pre> public class Foo { private SimpleDateFormat sdf = new SimpleDateFormatC'E MMM dd HH:mmss"); } </pre>
올바른 예제	<pre> public class Foo { private SimpleDateFormat sdf = new SimpleDateFormatC'E MMM dd HH:mmss",Locale.KOREA); } </pre>

	}
Rule-17	ImmutableField
설명	생성자에서 Assign된 변수를 Final로 선언하지 않았음
조치방법	생성자에서만 할당되는 변수라면 final로 선언
잘못된 예제	<pre> public class Foo { private int x; public Foo() { x = 7; } public void fooO { int a = x + 2; } } </pre>
올바른 예제	<pre> public class Foo { final int x; public Foo() { x 나 } public void fooO { int a = x ♦ 2; } } </pre>
Rufe-18	AssignmentToNonFinalStatic
설명	static 필드의 안전하지않은 사용 가능성
조치방법	final을 선언하여 막을것
잘못된 예제	<pre> public class StaticField { static int x; public FinalFields(int y) { x = y; // unsafe } } </pre>
올바른 예제	<pre> public class StaticField { static final int x; public FinalFieldspnt y) { x = y; // unsafe } } </pre>
Rule-19	AvoidSynchronizedAtMethodLevel

설명	method 레벨의 synchronization 보다 block 레벨 synchronization 을 사용하는 것이 바람직함 새로운 코드가 메소드에 추가되었을때, 메소드 레벨 동기화는 역효과를 일으킨다. »락레벨 동기화는 동기화가 필요한 코드가 동기화될 수 있도록 도울것이다.
조치방법	블럭레벨 snc 로 변경한다.
잘못된 예제	<pre>public class Foo { synchronized void fooO () }</pre>
올바른 예제	<pre>public class Foo { void barO { synchronized(this) { } } }</pre>
Rule-20	AbstractClassWithoutAbstractMethod
설명	Abstract Class내에 Abstract Method가 존재하지 않음
조치방법	비 추상 메소드들 제거하거나 abstract 임을 명시함것
잘못된 예제	<pre>public abstract class Foo { void int method 10 (-) void int method20, 습X }</pre>
올바른 예제	<pre>public abstract class Foo { abstract v6ki int method 1() {어 } abstract void int method20 (-) // consider using abstract methods or removing // the abstract modifier and adding protected constructors }</pre>
Rule-21	UncommentedEmptyMethod
설명	Empty Method 가 사용되었을, 의도적으로 Empty Method 를 구현했다면 그에 대항하는 주석을 기재하는 것이 바람직함.
조치방법	불필요할 경우 삭제, 의도적이면 주석추가
잘못된 예제	<pre>public void doSomethingO (}</pre>
올바른 예제	삭제
Rule-22	AvoidConstantsInterface
설명	Interface는 클래스의 behavior 을 구현하는 데에만 사용해야 함. 상수를 담는 컨테이너로 사용하는 것은 바람직하지 않»

	지금 상황에서는 클래스를 사용하는 것이 바람직함
조치방법	interface 내에 상수를 선언하지 않것. (상수클래스를 사용)
잘못된 예제	<pre>public interface Constantsinterface { public static final int CONSTANT1=0; public static final String CONSTANT2=T; }</pre>
올바른 예제	<pre>public class ConstantsClass { public static final int CONSTANT1=0; public static final String CONSTANT2=*r ; }</pre>

9.S 기타 코드 표준 개선 예제

Rule-23	Duplicateimports
설명	import문이 중복 선언 되었음
조치방법	중복 import 삭제
잘못된 예제	<pre>import java.lang.St ring; import java.lang.*; public class Foo 0</pre>
올바른 예제	<pre>import java.lang.*; public class Foo 0</pre>
Rule 24	ImportFromSamePackage
설명	동일 패키지에 있을 때는 import문을 사용탈 필요가 없음
조치방법	import 삭제
잘못된 예제	<pre>package foo; import foo.Buz; import foo?; public class Barf)</pre>
올바른 예제	<pre>package foo; public class BarQ</pre>
Rule-25	SystemPrintln
설명	개발 시 디버그 작업을 위해 사용한 System.outprintlnO문이 남아 있음. 불필요한 System.outprintlnOS^ 시스3에 부하를 발생 시킨다. (System.out 으로 시작하는 부분을 모두 찾도록 기본PMD를을 보완함)
조치방법	삭제 또는Logger 사용을 고려하기 바람
잘못된 예제	<pre>class Foo{ Logger log = Logger.getLogger(Foo.dass.getNameO) ; public void testA 0 {</pre>

	<pre> System.out.println("Entering test*"); } }) </pre>
올바른 예제	<pre> class Foo{ public void testA () { // Better use this debugf("Entering test"); } } </pre>
Rule-26	VariableNamingConventions
설명	다음 2가지 경우 중 하나임. • final 변수 이름이 대문자가 아님. • non-final 변수에 underscore 가 포함되어 있음.
조치방법	영명규칙준수 - final 대문자 - 일반변수는 . 제거
잘못된 예제	<pre> public class Foo { public static final int my.num = 0; public String my.Test = ""; DataModule dm_Test = new DataModuleO ; } </pre>
올바른 예제	<pre> public class Foo { public static final int MY.NUM = 0; public String myTest = ""; DataModule dmTest = new DataModuleO ; } </pre>
Rule-27	MisleadingVariableName
설명	non-field 이름이 m_ 으로 시작함. M- 은 field 를 지칭하는 것이므로 혼동을 줄 수 있음
조치방법	Rule준수 m_ 삭제
잘못된 예제	<pre> public class Foo { public void bar(String m_baz) { // Bad int m_boz = 42; // Bad } } </pre>
올바른 예제	<pre> public class Foo { public void bar(String baz) { // Bad int boz = 42; // Bad } } </pre>

	<pre> } } </pre>
Rule-28	AvoidArrayLoops
설명	배열의 값을 루프문 S 이용하여 복사하는 것 보다 System.arraycopyO 메소드풀 이용하여 복사하는 것이 효율적이며 수행 속도가 빠름
조치방법	루프를 이용한 배열값 복사 대신 System.arraycopy 메소드를 사용할것
잘못된 예제	<pre> public class Test { public void barO { into a = new int [10]; into b = new int[10]; for (int i=0;i<10;i++) { b[i]=a[i]; } } } </pre>
올바른 예제	<pre> public class Test { public void bar() { into a = new int(10); into b = new int[10]; System.arraycopy(a, 0, b, 0, 10); } } </pre>
Rule-29	UnnecessaryWrapperObjectCreation
설명	불필요한 Wrapper Object가 생성되었음. Parsing 메소드 1 직접적 호출하는 것이 바람직함.
조치방법	parseXXX 형태의 메소드로 변경
잘못된 예제	<pre> public int convert(String s) { int i, i2; i = Integer.valueOf(s).intValue(); // this wastes an object i2 = Integer.valueOf(i).intValue(); // this wastes an object return i2; } </pre>
올바른 예제	<pre> public int convert(String s) { int i, i2; i = Integer.parseInt(s); // this is better i2 = i; // this is better return i2; } </pre>

Rule-30	AvoidThrowingRawExceptionTypes
설명	가공되지 않은 Exception을 throw하는 것은 비추천
조치방법	사용자 정의 Exception을 사용하도록 함
«못된 예제	<pre>public class Foo { public void barO throws Exception { throw new ExceptionO ; } }</pre>
올바른 예제	<pre>public class Foo (public void barO throws BizException { throw new BizExceptionO ; } }</pre>
Rule-31	AvoidThrowingNullPointerException
설명	NullPointerExceptionS throw하는 것은 비추천
조치방법	대신 IllegalArgumentException을 사용할것
»못된 예제	<pre>public class Foo { void barO { throw new NullPointerExceptionO ; } }</pre>
올바른 예제	<pre>public class Foo { void barO { throw new UlegalArgumentExceptionO ; } }</pre>
Rule-32	Stringinstantiation
설명	필요없는 Instance가 생성되어 있음
조치방법	new로 생성하지 말고, 직접 문자형에 대입 ex) String bar = *bar ^M ;
잘못된 예제	<pre>public class Foo { private String bar = new StringCbar); }</pre>
올바른 예제	<pre>public class Foo { String bar = "bar"; }</pre>
Rule-33	StringToString
설명	String 객체에서 toStringO함수콜 사용하는 것은 불필요함
조치방법	toString 삭제

잘못된 예제	<pre> public class Foo { private String bazO { String bar = "howdy"^M; return bar.toStringO ; } } J </pre>
올바른 예제	<pre> public class Foo (private String bazO I String bar = "howdy"; return bar; }) </pre>
Rule-34	1 nefficientStringBufferi ng
설명	String Buffer 함수에서 nonliteral 을 직접 concatenate 하지 말 것. Nonliteral conatenation 는 별도로 처리할 것.
조치방법	append를 사용하여 처리할것
잘못된 예제	<pre> public class Foo { void barQ { StringBuffer sb=new StringBuffer(**tmp = **System.getPropertyC*java.io.tmpdir^M)); } } >1 ' </pre>
올바른 예제	<pre> public class Foo { void barO { StringBuffer sb = new StringBuffer(^Mtmp = **); sb.append(System.getProperty(^Mjava.io.tmpdir")); } } </pre>
Rule-35	InefficientEmptyStringCheck
설명	Empty String 을 체크하기 위해 String.trimO.length() 을 사용하는 것은 피하도록 함. Character.isWhitespaceO를 사용하는 것이 바람직함
조치방법	isWhitespace를 이용하여 공백을 체크하는 로직으로 변경처리
잘못된 예제	<pre> public class Foo { void bar(String string) { if (string != null && string.trimO sizeO > 0) { doSomethingO ; } } }) </pre>

올바른 예제	<pre>import com.sli.framework.util.StringUtil; public class Foo { void bar(String string) { if (!StringUtil.isEmpty(string)) { doSomethingO; } } }</pre>
Rule-36	UselessStringValueOf
설명	String 을 append 합 경우, String.valueOf 함수를 사용할 필요 없음. ValueofO 대신 직접 사용하는 것이 바람직함.
조치방법	valueOf 제거
틀린 예제	<pre>public String convert(int i) { String s; s = "a" + String.valueOf(i); return s; }</pre>
올바른 예제	<pre>public String convert(int i) { String s; s += "a"; return s; }</pre>
Rule-37	UnusedPrivateField
성명	사용되지 않는 Private field가 선언되었음
조치방법	field 삭제
탈뭇된 예제	<pre>public class Something { private static int FOO = 2; // Unused private int i = 5; // Unused private int j = 6; public int addOneO { return j++; } }</pre>
올바른 예제	<pre>public class Something { private int j = 6; public int addOneO { return j++; } }</pre>

Rule-38	Unused PrivateMethod
설명	사용되지 않는 Private Method가 선언되었음
조치방법	method 삭제
잘못된 예제	<pre>public class Something { private void fooO 0 // unused }</pre>
올바른 예제	<pre>public class Something ()</pre>
Rule-39	UnusedFormalParameter
설명	사용되지 않는 파라미터가 있음
조치방법	파라미터 삭제
»못된 예제	<pre>public class Foo { private void bar(String howdy) { // howdy is not used } }</pre>
올바른 예제	<pre>public class Foo { private void barO { // howdy is not used } }</pre>