Glue Framework 4

v1.2.1

POSCO ICT

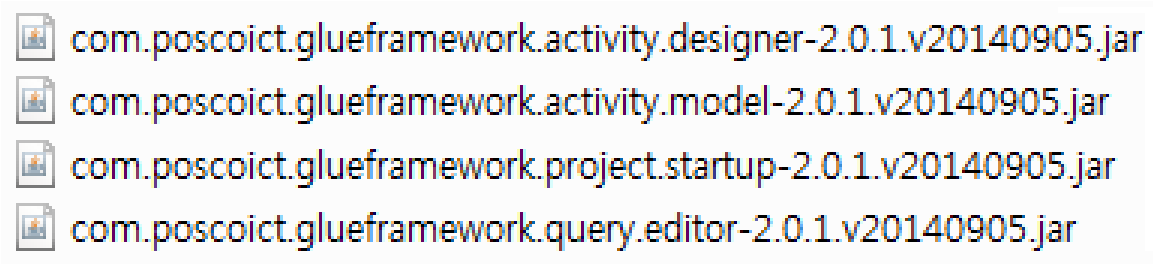
- 목차 -

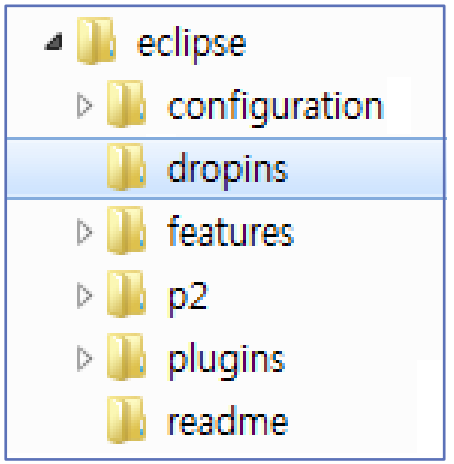
Ⅰ. Glue Framework 개요

- **URLs**
  - **http://www.solutionpot.co.kr/ (LM포털)**
    - GlueSDK, plugin 다운로드
      - 공지사항, Q&A, FAQ, 다운로드 게시판 제공
  - **http://www.solutionpot.co.kr/doc/**
    - 온라인 문서 / demo 제공
  - **http://www.oracle.com**
    - Java SE 를 다운로드
  - **http://eclipse.org**
    - Eclipse IDE for Java Developers 를 다운로드
  - **http://tomcat.apache.org**
    - Tomcat 7.0 다운로드
  - **http://maven.apache.org**
    - Maven 다운로드
  - **http://logback.qos.ch**
    - logback 다운로드

- **Eclipse**
  - **기본 개발 IDE**
    - Eclipse IDE for Java EE Developers
    - Eclipse IDE for Java Developers
      - Juno(v4.2) 이후 버전 권장
  - **plug in 기능**
    - Glue Project
    - Glue Activity Diagram
    - Glue Query Editor
  - **Glue SDK**
    - Application 개발에 필요한 기본 Kit.
    - Glue framework library
    - Java Doc
    - Sample Template
    - 제공 plug in 에서 사용되는 config 파일

- **Eclipse – plug in**
  - **Glue Plug-in 2 는 LGPL license에 따라 소스 공개함.**
  - **Glue의 패키지는 major, minor, patch 형태로 버전 관리함.**
    - Glue패키지로 Glue Framework, GlueMaster, Glue Plug-in이 있음.
    - 이후 Glue Framework 4 와 같이 패키지 이름에 major 버전을 같이 표현함.
  - **Glue Framework 4를 이용한 Application 개발은 Glue Plug-in 2를 필요로 함.**
    - Glue Framework 3은 Glue Plug-in 1을 필요로 함.
    - IDE(Eclipse)별로 2개 major 버전의 Glue Plug-in 설치 불가함.
  - **plug in 파일은 name_version.일자.jar 임.**
    - 예 : xxx_2.0.0.v20130805.jar
  - **dropins 폴더에 위치.**
    - plugins 폴더에 두어도 무방하나,
      plug in 재설치 시 해당 plug in을 찾기 어려움

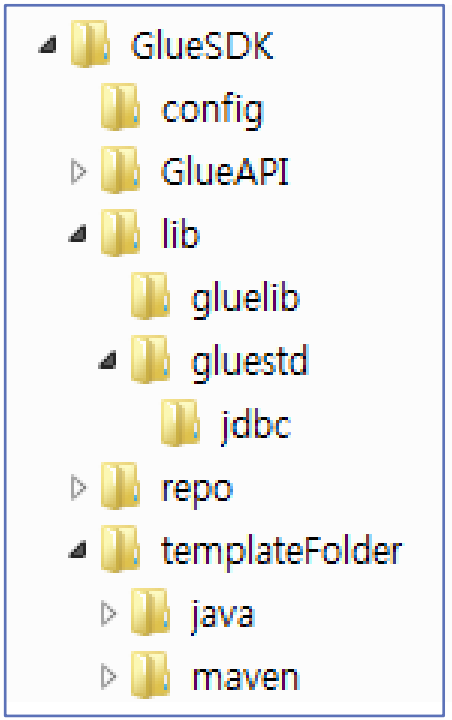




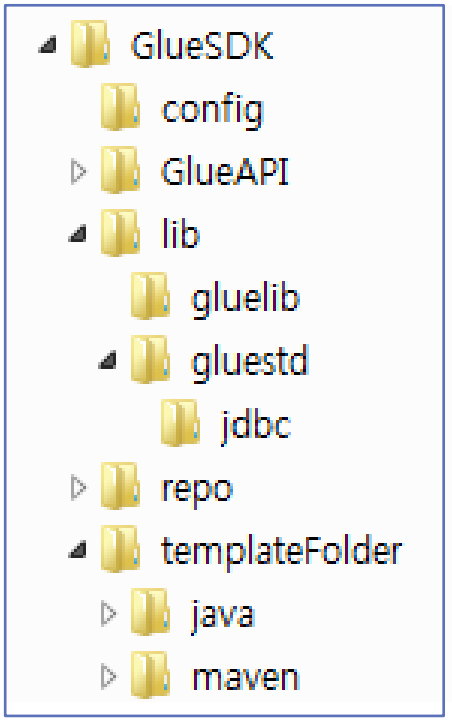
- **Eclipse – plug in**
  - **plug in 이 재배포 되는 경우 –clean 옵션이 적용된 바로 가기 이용.**
    - plug in 패치/업그레이드 시 기존 plug in을 지우고 재 배포된 plug in을 설치(복사)한다.
    - plug in 이 바뀌면 Eclipse를 재 실행 해야 하며, –clean 옵션이 적용된 바로 가기를 이용한다.

  - **Glue Plug-in 재설치 과정**
    1. 실행 중인 Eclipse 종료
    2. 기존 plugin 삭제
    3. 새로운 plugin 복사
    4. '-clean'이 포함된 바로 가기 실행
    5. GlueSDK SDK Location 재 설정
       (Window → Preferences → Glue Framework)

- **Eclipse – Glue SDK**
  - **Application 개발에 필요한 기본 Kit.**
    - GlueSDK는 그림과 같은 구조임.
    - GlueSDK의 내용은 개발자가 개별로 설정하지 않고, PL의 의해 동일 설정이 유지되기를 권장.
      - GlueSDK의 구성물 중 상용 License, LGPL 를 따르는 것은 직접 추가 해야 함. (ex. ojdbc.jar 등)
    - GlueSDK의 구성물 중 Glue Framework 4는 GPL 2를 따름.
      - <GlueSDK>/lib/gluelib
      - <GlueSDK>/repo
    - Glue Framework 패치/업그레이드 시 GlueSDK의 구성물 중 gluelib(repo)가 대상임.
    - Eclipse IDE에서 GlueSDK의 위치를 설정하면 Glue Plugin은 GlueSDK를 사용함.
    - Eclipse IDE의 workspace 별로 GlueSDK를 지정함.
    - 서버 환경 구성 시에는 lib 폴더의 jar 파일을 활용함.



- **Eclipse – Glue SDK**
  - **config**
    - config 폴더에는 glue-config.xml 파일이 있으며 Glue Plugin에서 사용하는 설정 정보 관리
  - **GlueAPI**
    - 프로젝트 정보 (의존물/라이선스 등) 문서화.
    - lib/gluelib 에 존재하는 library(프로젝트 모듈)들의 java doc
  - **lib**
    - gluelib 와 gluestd로 구성.
    - gluelib 에는 프로젝트 모듈 결과물(glue library, jar 파일)이 위치함.
    - gluestd 에는 프로젝트 모듈에서 참고하는 의존 library들이 위치함. 즉, Application 실행시 필요한 모든 의존 library들이 위치함.
    - gluestd/jdbc 에는 Glue Plugin 중 Query Editor가 사용함.
  - **repo**
    - maven 지원용임
  - **TemplateFolder**
    - java 와 maven 으로 구성. 2가지 Type의 Sample Template.
    - Glue Project 생성시 사용되며, Sample Web Application을 포함함.
    - Java Project / Maven Project 지원



- glue-config.xml
  - activities 와 common 으로 구성.
  - glue-schema-4.x.x.jar 의 Schema 참고
    - schemaorg_apache_xmlbeans / src / GluePluginConfig.xsd

glue-config.xml

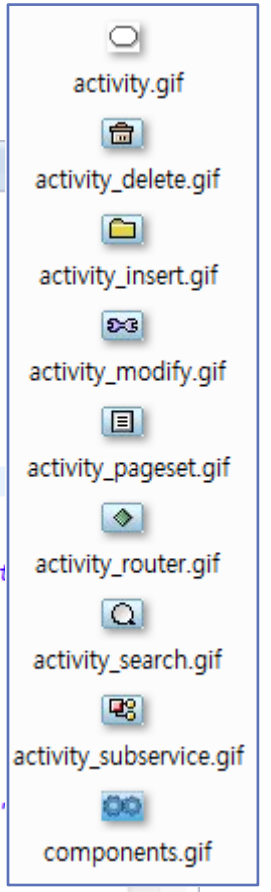
```
<?xml version="1.0" encoding="UTF-8"?>
<glue-eclise-config
    xmlns="http://www.poscoict.com/glueframework/plugin/config"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.poscoict.com/glueframework/plugin/config
                        GluePluginConfig.xsd ">
    <activities>
        <activity name="Router" icon="icons/activity_router.gif" addSeperator="true">
        <activity name="SubService" icon="icons/activity_subservice.gif">
        <activity name="Jdbc Search" icon="icons/activity_search.gif">
        <activity name="Jdbc Insert" icon="icons/activity_insert.gif">
        <activity name="Jdbc Modify" icon="icons/activity_modify.gif">
        <activity name="Jdbc Delete" icon="icons/activity_delete.gif">
        <activity name="Message Parse" icon="icons/components.gif" addSeperator="true">
        <activity name="Message Create" icon="icons/components.gif">
    </activities>
    <common>
        <document>
        <service>
        <query>
        <class-generator>
        <activity-template>
    </common>
</glue-eclise-config>
```

Design Source

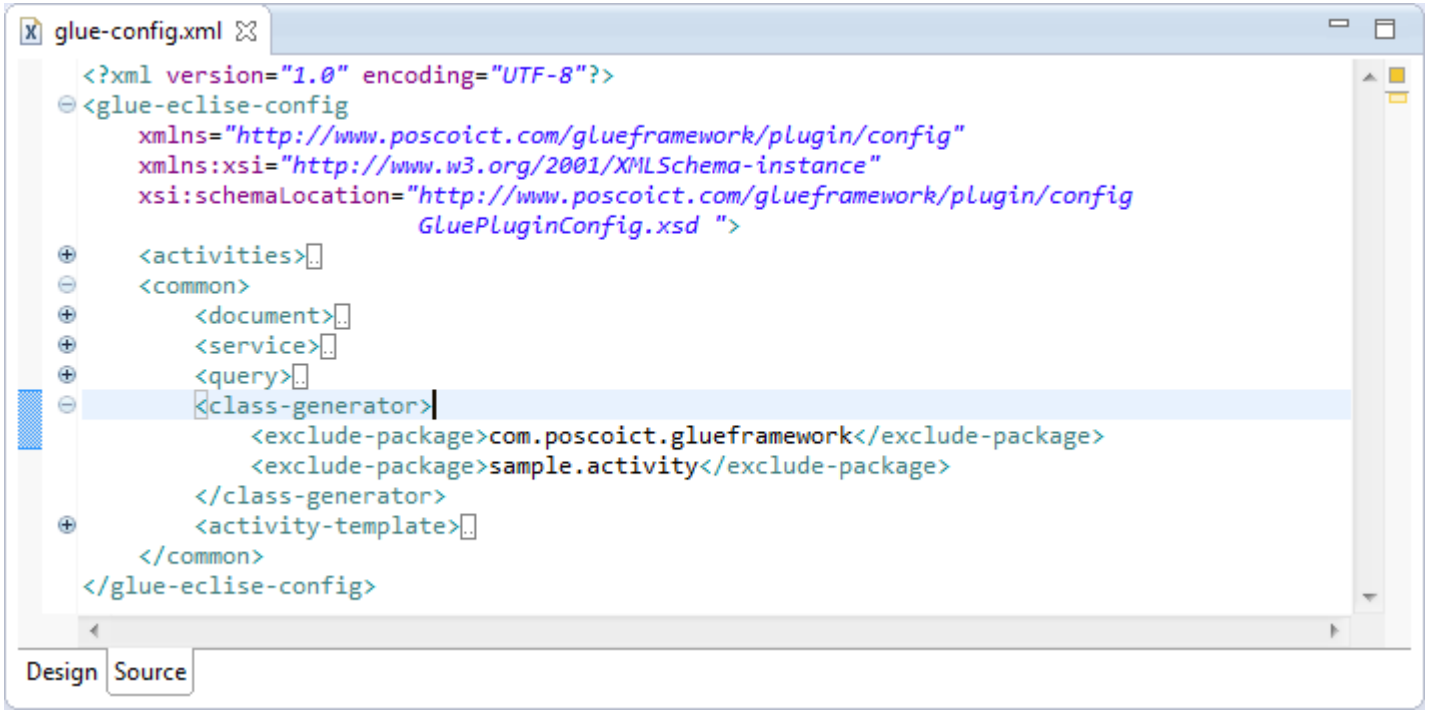
- **glue-config.xml**
  - **activities 영역**
    - Reuse Activity 관리
    - icon : icons/파일명 으로 정의함. 허용 파일명은 그림 참고.
    - addSeparator : Palette의 구분선을 정의. Reuse Activity의 Grouping 용도.
    - color : 허용 color는 schema 참고.

glue-config.xml

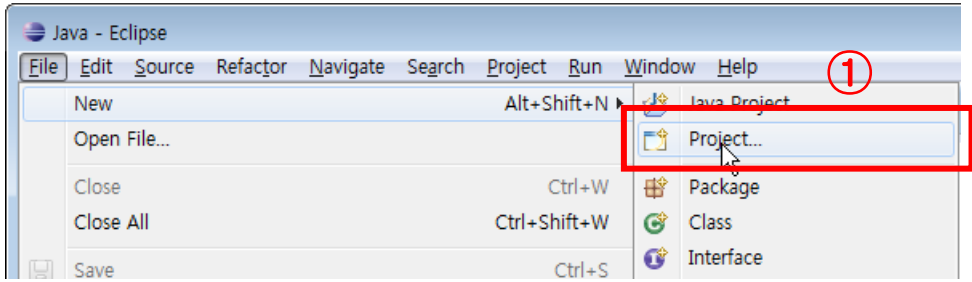
```
<?xml version="1.0" encoding="UTF-8"?>
<glue-eclise-config
    xmlns="http://www.poscoict.com/glueframework/plugin/config"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.poscoict.com/glueframework/plugin/config
                        GluePluginConfig.xsd ">
    <activities>
        <activity name="Router" icon="icons/activity_router.gif" addSeperator="true">
            <bubble-tip>Default Router Activity</bubble-tip>
            <template>Glue Router template</template>
            <color>green</color>
            <default-properties>
                <property name="class" value="com.poscoict.glueframework.biz.activity.GlueDefaultRout
            </default-properties>
        </activity>
        <activity name="SubService" icon="icons/activity_subservice.gif">
            <bubble-tip>SubService Activity</bubble-tip>
            <template>Glue SubService template</template>
            <color>lightBlue</color>
            <default-properties>
                <property name="class" value="com.poscoict.glueframework.biz.activity.GlueSubService'
                <property name="service-name" value="sample-service" />
                <property name="new-transaction" value="false" />
            </default-properties>
```

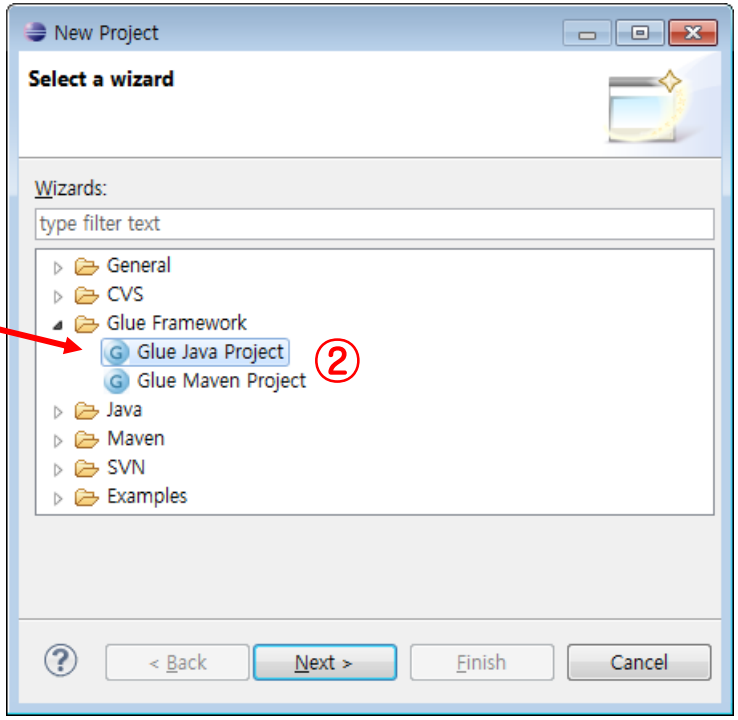


- **glue-config.xml**
  - **common 영역**
    - document : 서비스 명세서 관련 설정.
    - service : GlueService 파일의 인코딩 유형 정의
    - query : GlueQueryEditor 파일의 인코딩 유형 정의
    - class-generator : Activity Diagram의 Class 생성 제외 대상 관리
    - activity-template : Custom Activity의 template 코드



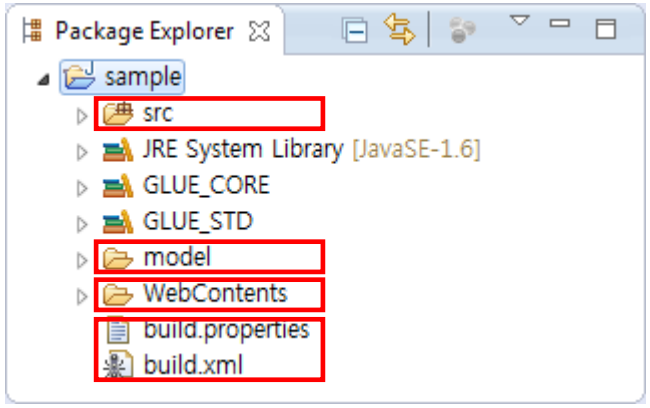
- 기능 1 : Glue Project
  - Glue Project 기능
    - 자동으로 개발환경 설정함.
      - Glue F/W으로 개발에 필요한 Lib Path 설정
      - Glue F/W의 환경 파일 자동 생성
        - applicationContext.xml, log4j.xml 등
    - 자동 Build 환경 구성
      - Build.xml으로 자동 Build 환경 구성
    - Sample Application 예제 제공
  - Glue Project 사용 방법





기능 1 : Glue Project

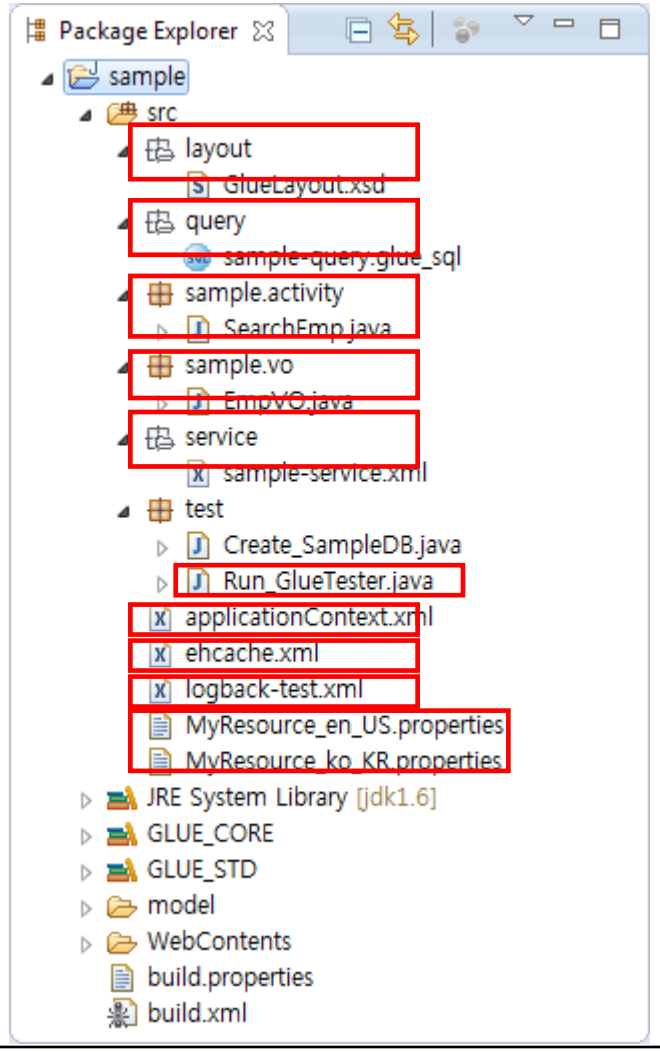
Java Project의 Folder 설명



이름	설명
Project명	Java Type 의 Project 명
src 폴더	소스 폴더로 다음과 같은 파일이 위치한다. - Bean 정의 파일 : applicationContext.xml - 캐시설정파일 : ehcache.xml - 로그 설정파일 : logback-test.xml - property 파일 : *.properties - Glue Query 파일 : query 폴더의 *-query.glue_sql - Glue Service 파일 : service 폴더의 *-service.xml - Java 파일(Activity, VO등) - GlueTester : test package 를 사용함.
model 폴더	설계 산출물 관리 Glue Activity Diagram 파일 : *.glue_uml_ad
WebContents 폴더	Web 소스 폴더로 다음과 같은 파일이 위치한다. - web.xml - dispacher-servlet.xml - JSP 파일 : *.jsp - js, css, html, gif, jpeg 등의 파일
ANT Build	Java 기반 Build Tool. War 생성 task 포함.

기능 1 : Glue Project

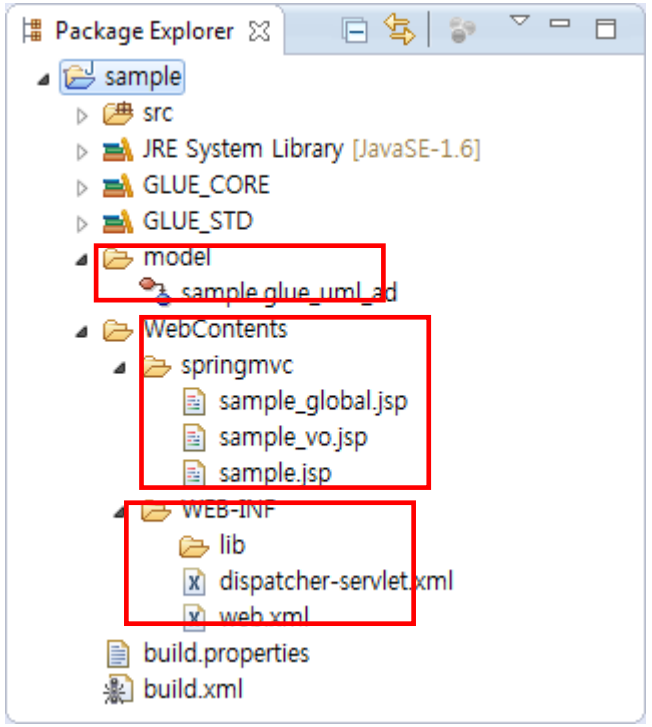
Java Project의 Folder 설명



이름	설명
layout 폴더	-msg.xml 파일의 위치. Layout 설계 산출물 XML
query 폴더	-query.glue_sql(또는 –hquery.glue_sql) 파일의 위치. SQL Query 설계 산출물 XML
sample.activity package	Custom Activity 예제. 개발 시 참조하여 개발할 Sample
sample.vo package	VO 예제 개발 시 참조하여 개발할 Sample
service 폴더	-serive.xml 파일의 위치. BusinessLogic 설계산출물 XML
Run_GlueTester.java	GlueTester 실행 소스. Local에서 개발 내용 Test 지원 Tool
applicationContext.xml	사용 컴포넌트(bean) 정의 . DB URL, Transaction Manager, DAO, ServiceManager, CacheManager, LayoutManager 등 정의
ehcache.xml	Memory Cache 정의
logback-test.xml	Logging 정의, Path 변경 필요
properties	Global 지원 또는 자체 Property 파일

기능 1 : Glue Project

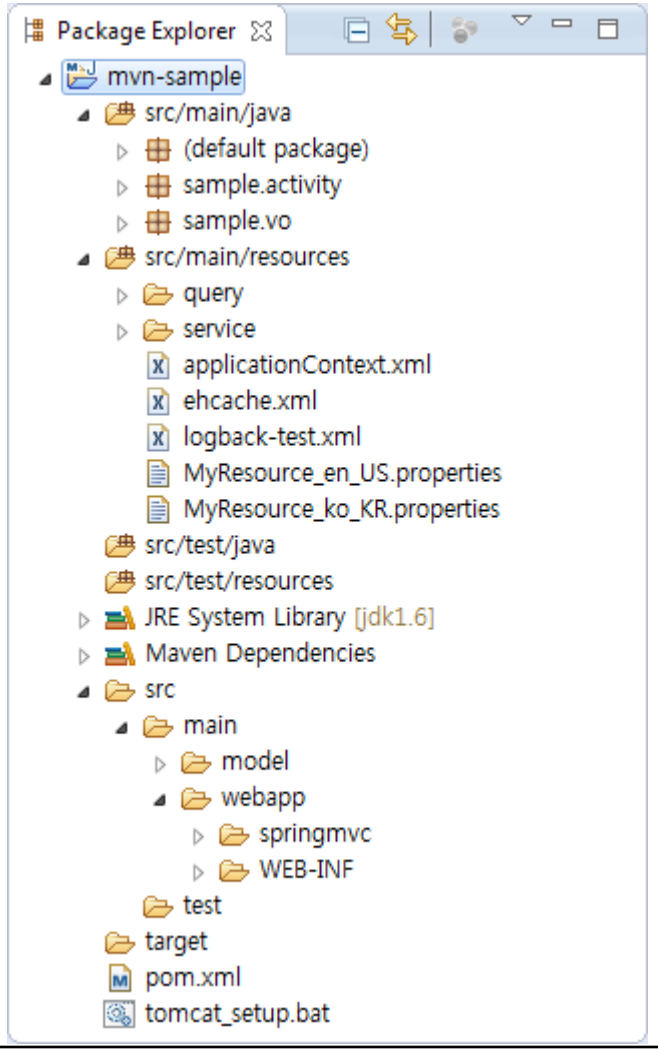
Java Project의 Folder 설명



이름	설명
model 폴더	.glue_uml_ad 파일의 위치 표준 설계서 작성
springmvc 폴더	JSP 파일의 위치. JSP 파일(HTML, JS, CSS 등 포함)은 WEB-INF를 제외한 WebContents 나 WebContents 하위 디렉토리에 위치함.
sample_global.jsp	다국어 적용 예제
sample_vo.jsp	VO 사용 화면 예제
sample.jsp	기본 화면 예제
WEB-INF 폴더	Web Application의 필수 폴더. Web Application을 위한 설정 파일이 위치함.
WEB-INF/lib 폴더	Library 폴더
dispatcher-servlet.xml	SpringMVC 의 DispatcherServlet이 필요로 하는 파일로 Spring Web MVC에 특화된 Component(bean)들을 포함함.
web.xml	Web Application의 필수 파일. servlet, filter, listener등 정의

기능 1 : Glue Project

Maven Project의 Folder 설명



이름	설명
src/main/java	배포대상 Java 소스의 위치
src/main/resource	배포대상 Java 소스를 제외한 리소스의 위치
src/test/java	테스트용 Java 소스 위치
src/test/resource	테스트용 Java 소스를 제외한 리소스의 위치
src/main/model	Activity Diagram 등의 설계산출물
src/main/webapp	Web 소스 폴더 WebContens 와 동일한 역할
pom.xml	maven 용 POM

기능 2 : Glue Query Editor

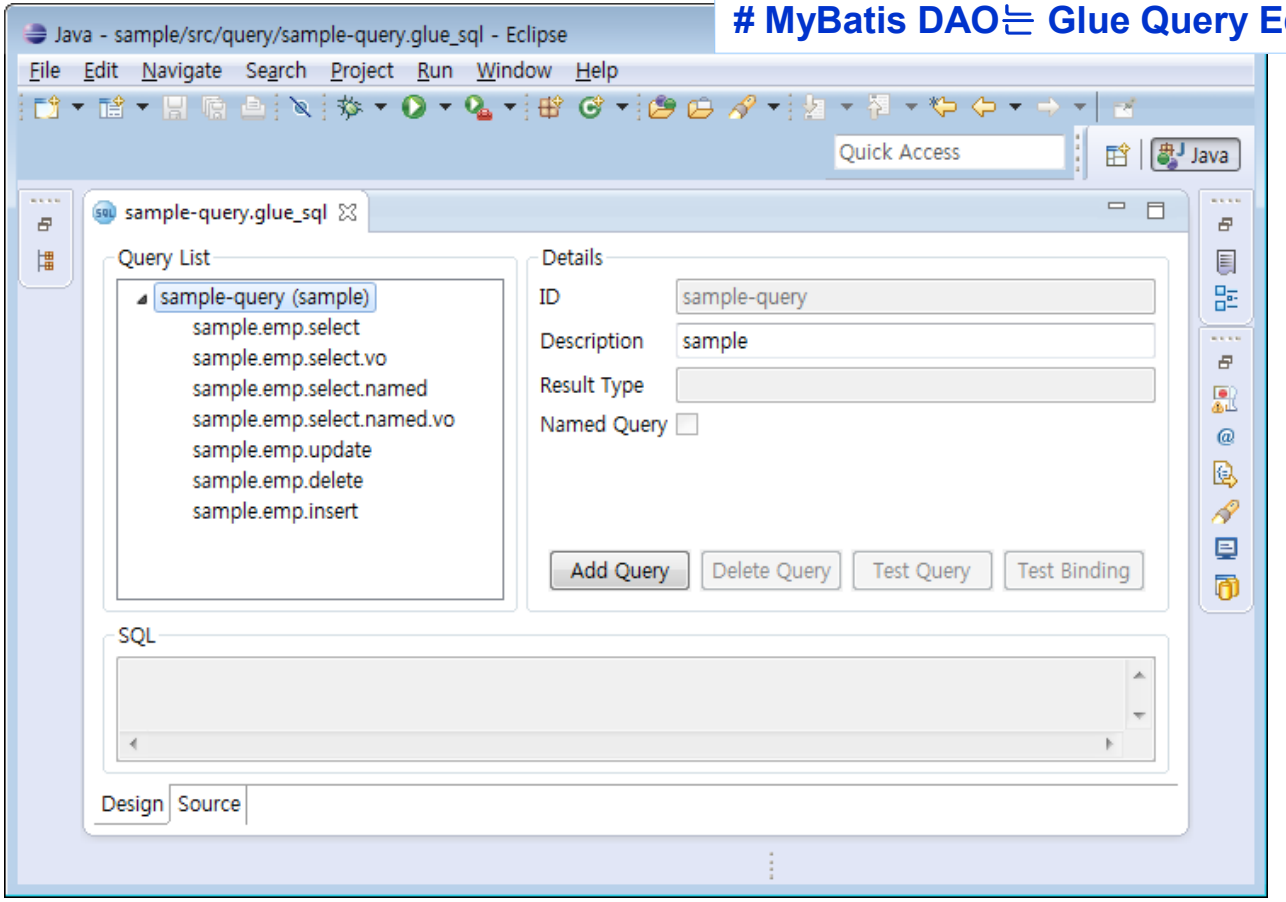
Glue Query Editor 기능

- Query 파일 생성 및 관리
- 파일명 : <name>-query.glue_sql

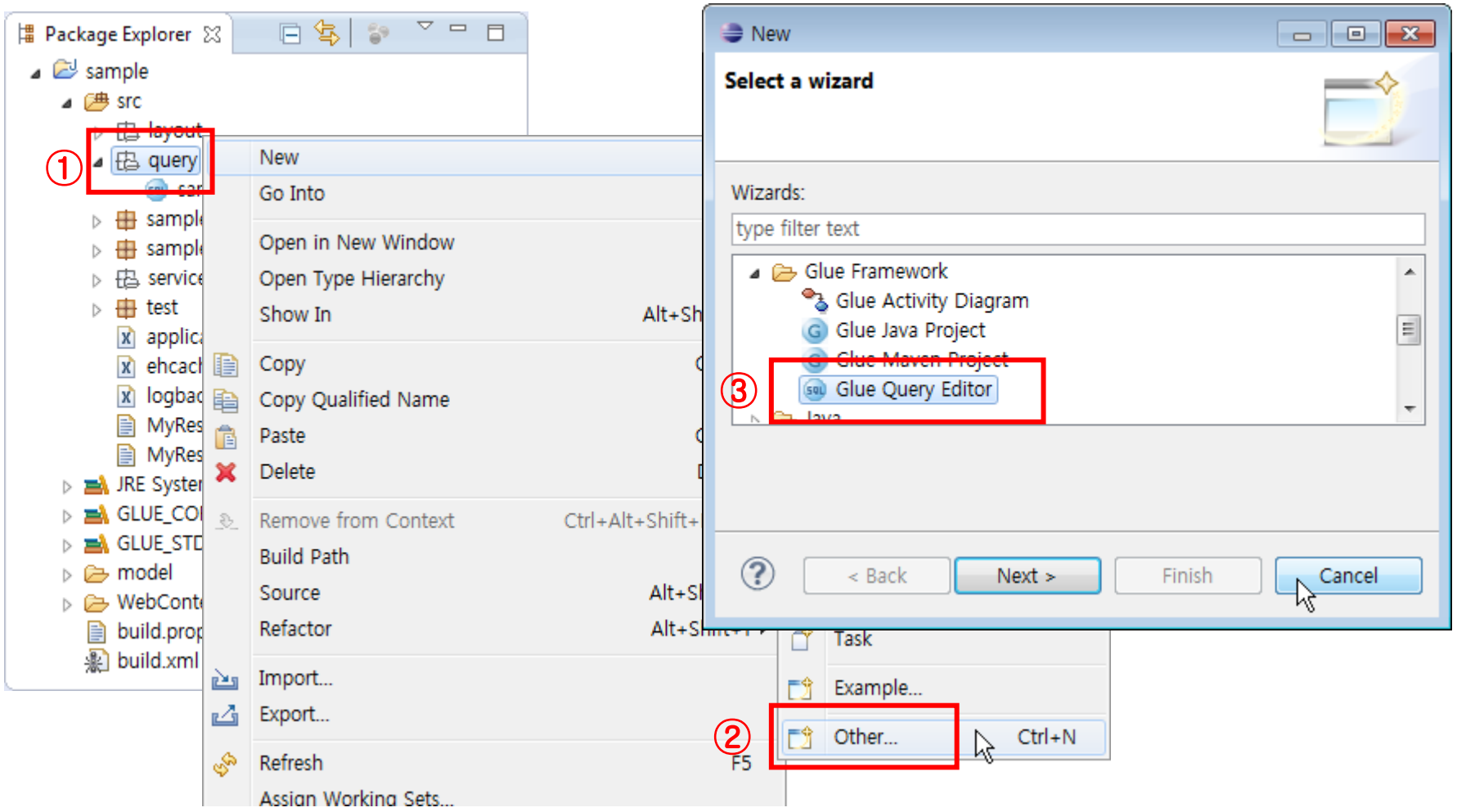
파일명 Naming Rule

1. name-query.glue_sql (JdbcDao 이용시)
2. name-hquery.glue_sql (HibernateDao 이용시)

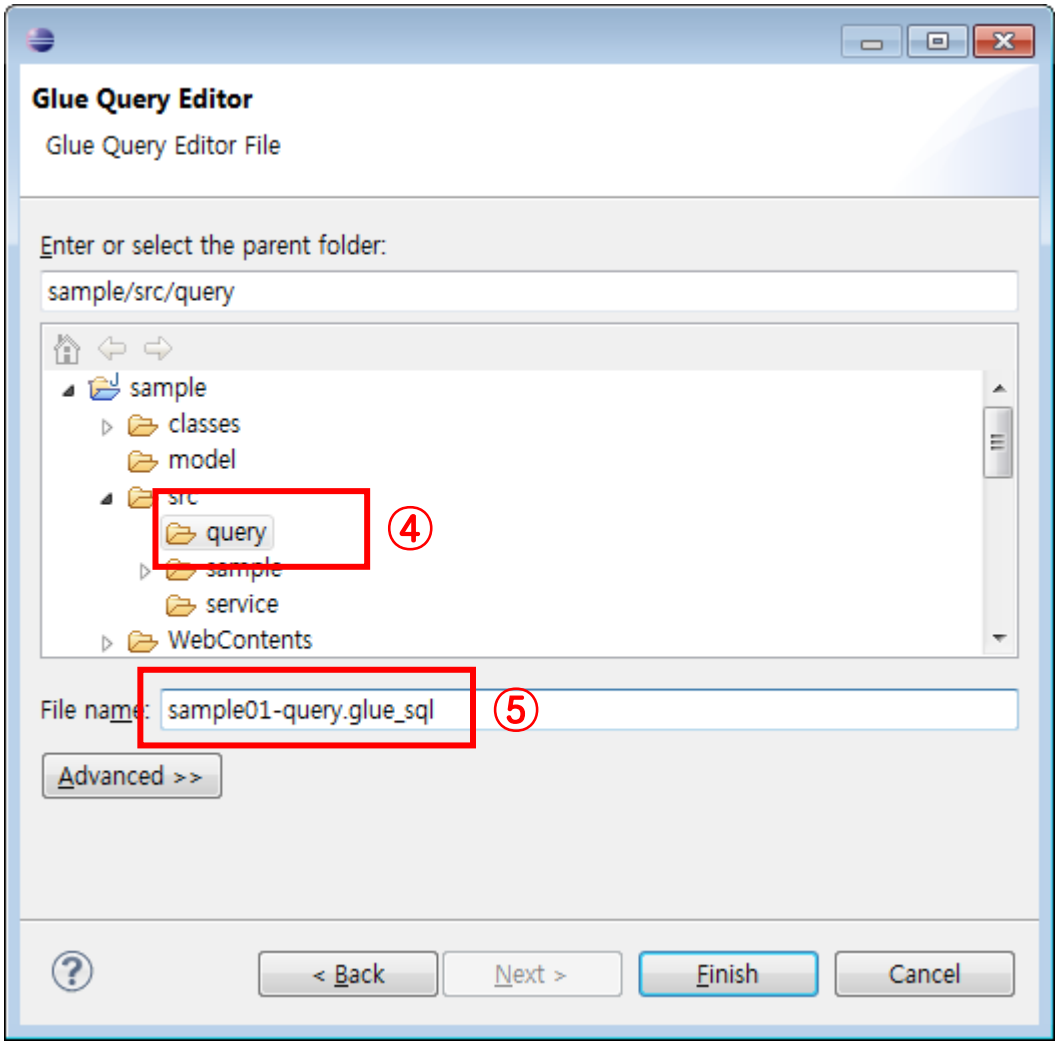
 # MyBatis DAO는 Glue Query Editor 불필요.



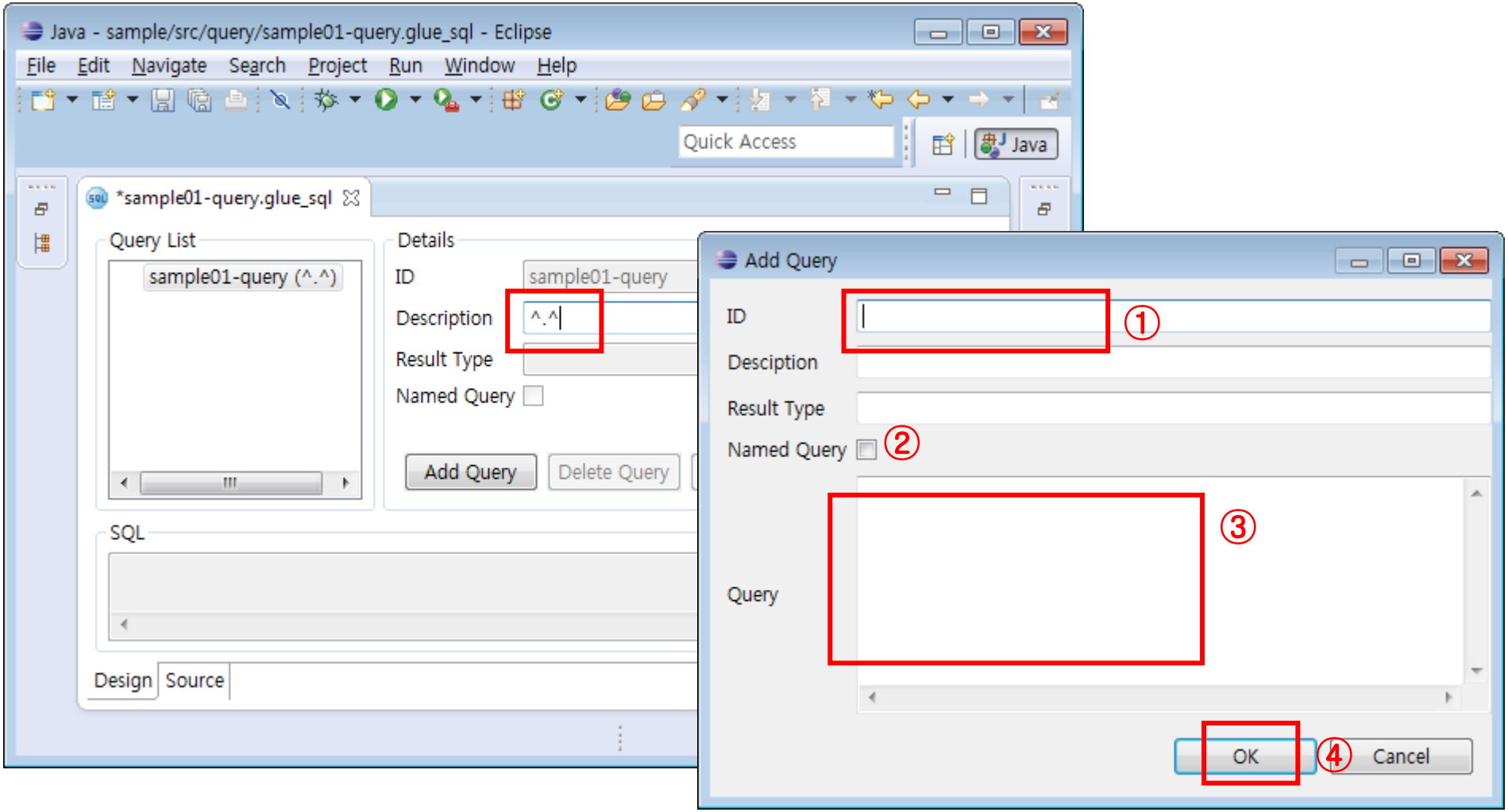
- 기능 2 : Glue Query Editor
  - Query 파일 생성
    - wizard 실행.



- 기능 2 : Glue Query Editor
  - Query 파일 생성
    - wizard 실행
      - ④ query 폴더 선택.
    - File name 수정
      - ⑤ name-query.glue_sql



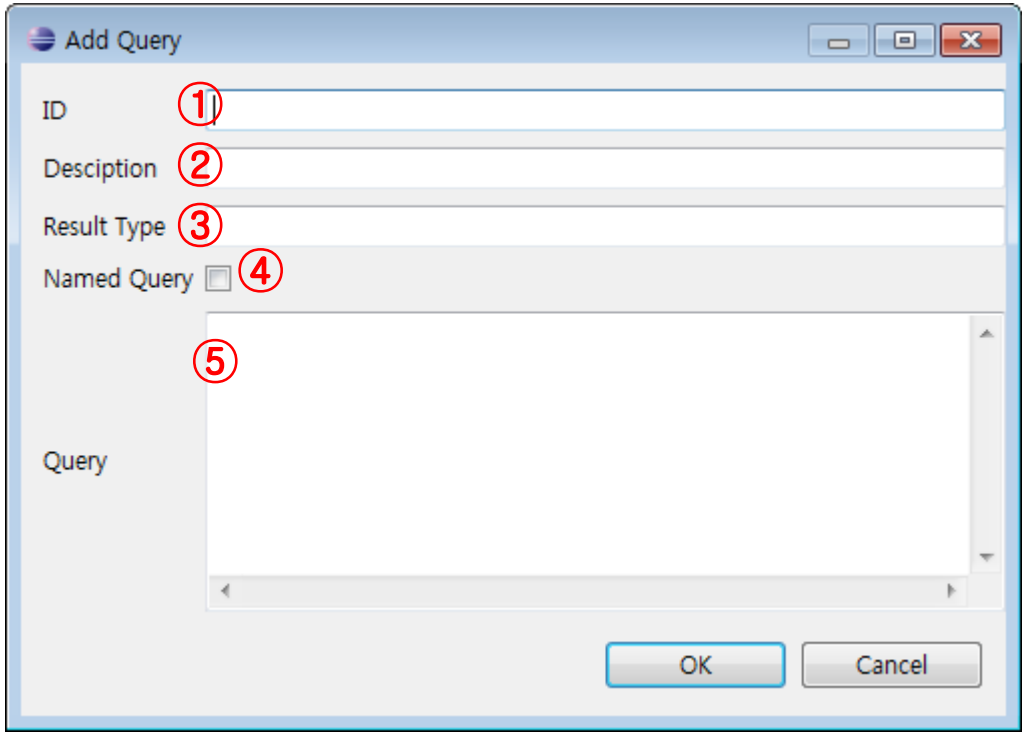
- 기능 2 : Glue Query Editor
  - Query 파일에 Query 추가



기능 2 : Glue Query Editor

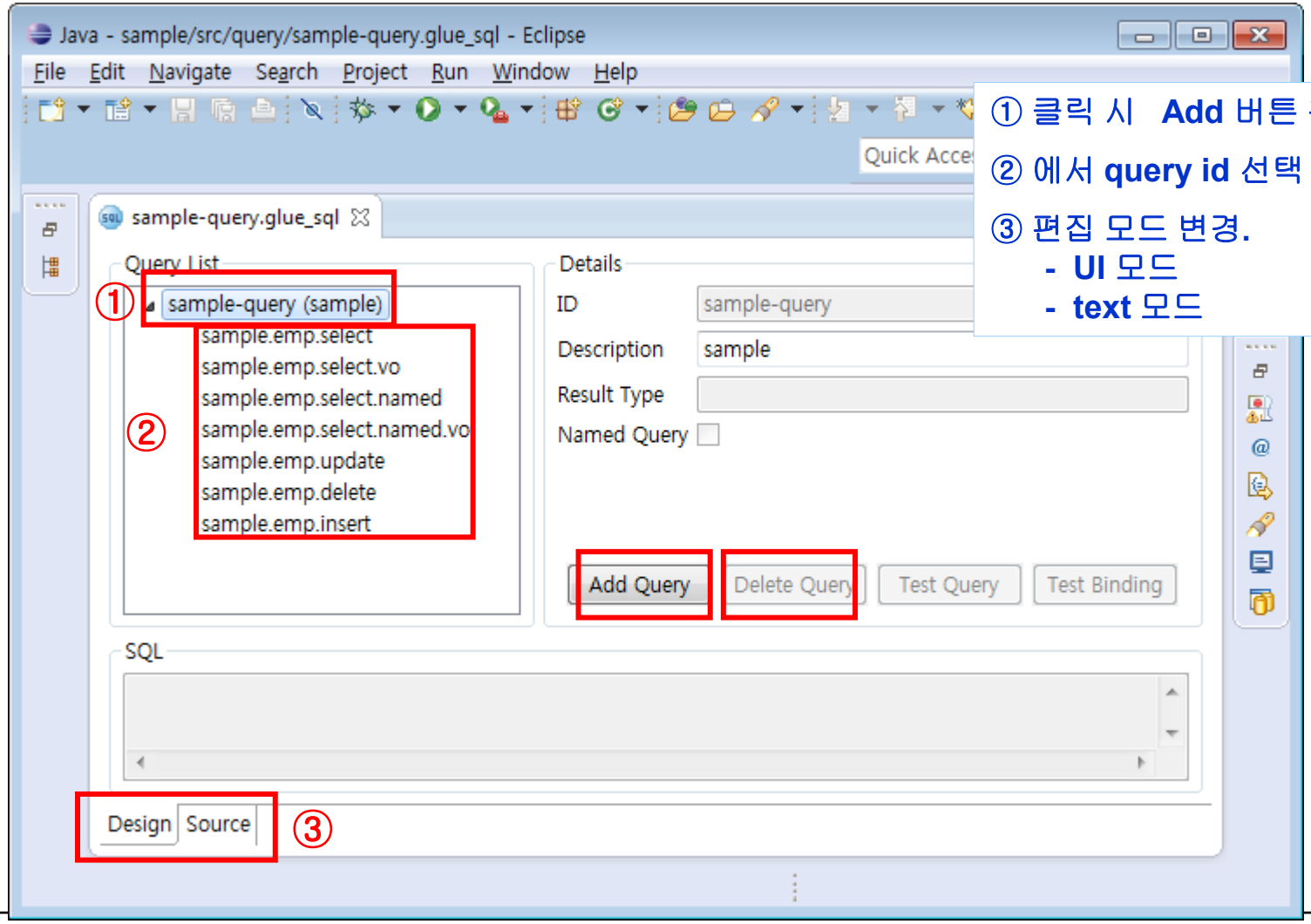
Query 파일(-query.glue_sql)에는 SQL만 관리한다.

- ① ID
  - 필수
  - 전체 Query 파일에서 유일
- ② description
  - 선택
  - 해당 SQL 설명 작성
- ③ Result Type
  - 선택
  - select query 의 경우
    1개 row를 표현하는 객체 type
  - package 포함 class 명
- ④ Named Query
  - 필수
  - 표준 SQL 인지 Named SQL인지
    - select * from emp where deptno=?
    - select * from emp where deptno=:key
- ⑤ query
  - 필수



기능 2 : Glue Query Editor

Query Editor 사용 방법



① 클릭 시 **Add** 버튼 활성화.

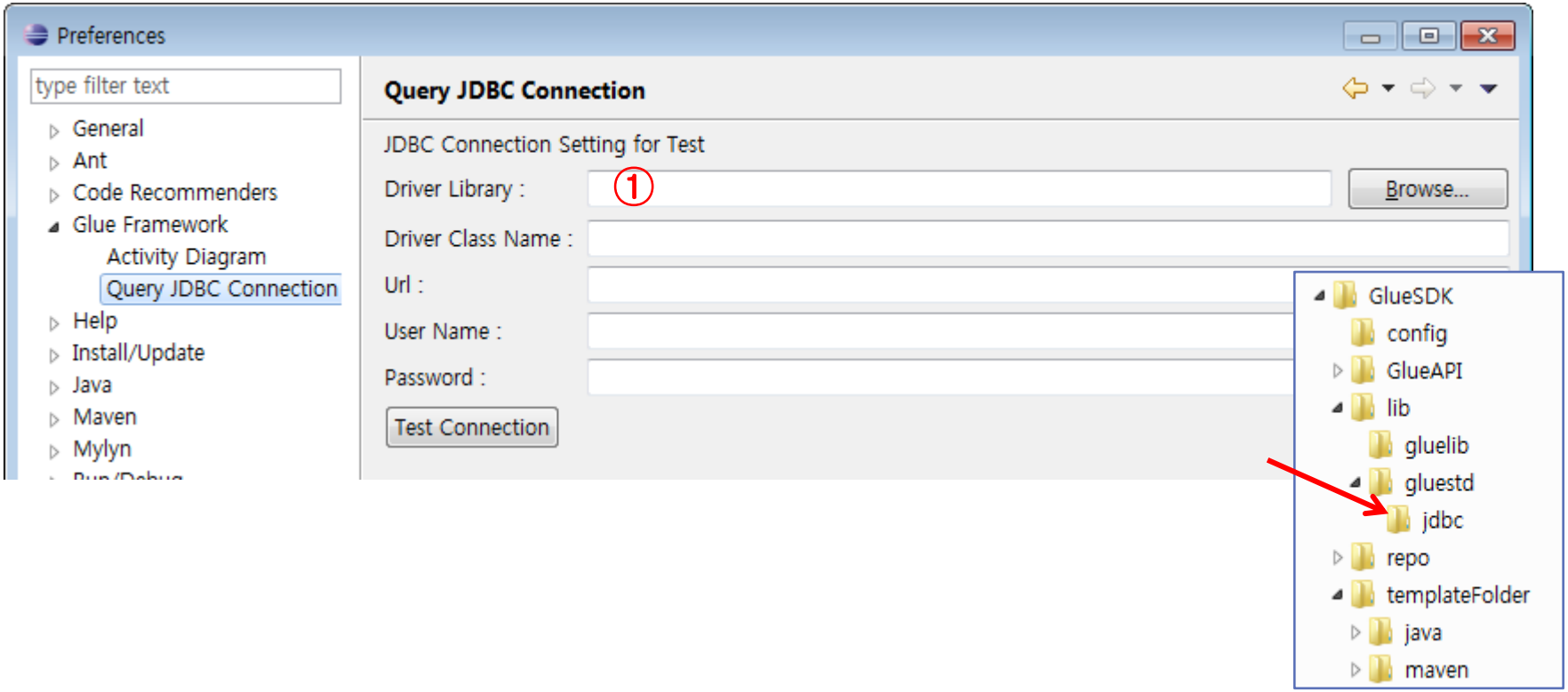
② 에서 **query id** 선택 시 수정/삭제 가능

③ 편집 모드 변경.
- **UI** 모드
- **text** 모드

기능 2 : Glue Query Editor

Query 테스트

- '?'를 포함하는 조회용 SQL의 테스트 지원
- SQL 테스트의 DB 연결정보는 Eclipse Workspace 별로 관리됨.
- Query 테스트시의 DB 연결정보는 ApplicationContext.xml 의 설정과 무관함.
- Driver Library 파일은 GlueSDK/lib/gluestd 외의 위치도 가능함.

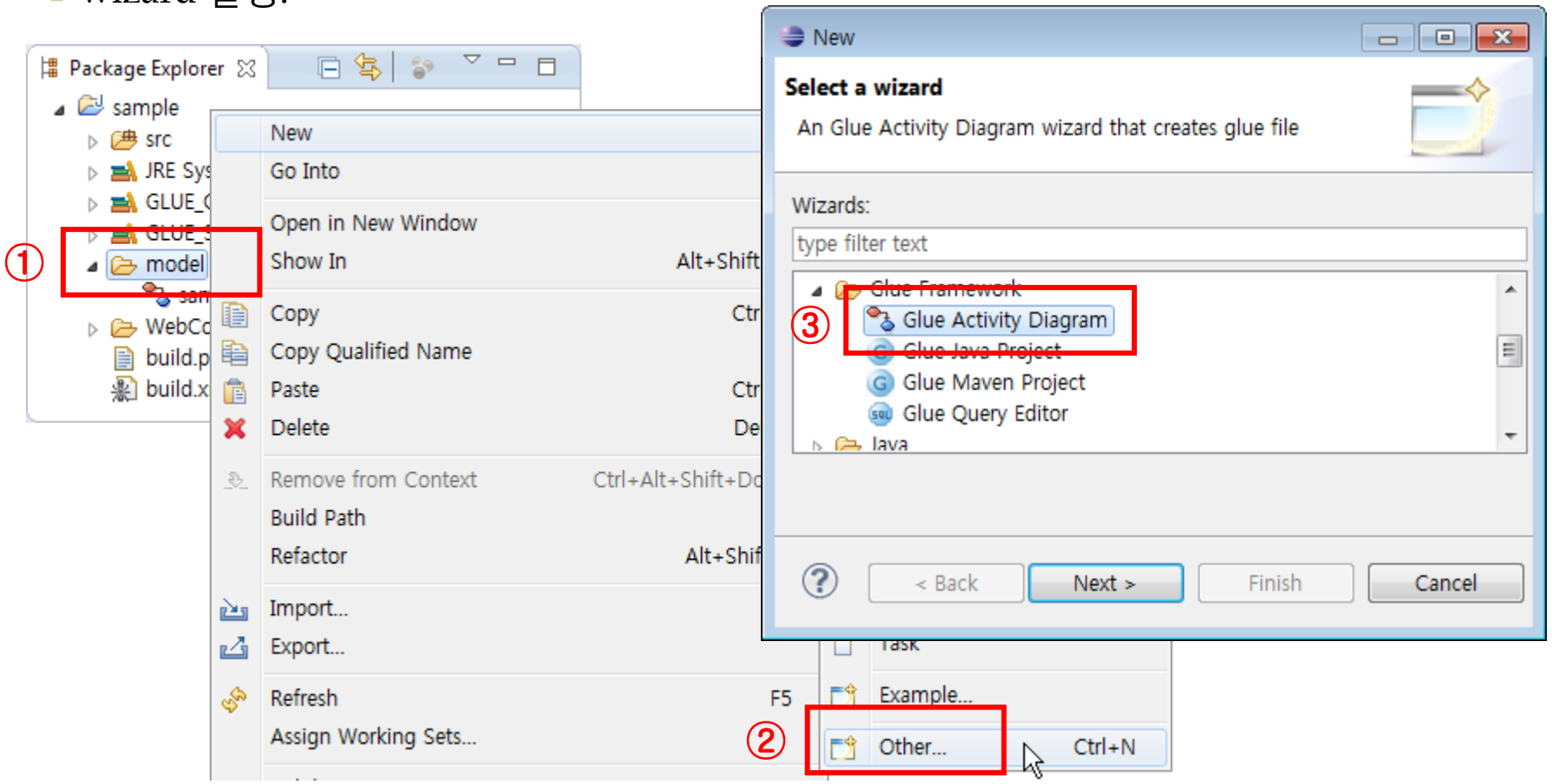


기능 3 : Glue Activity Diagram

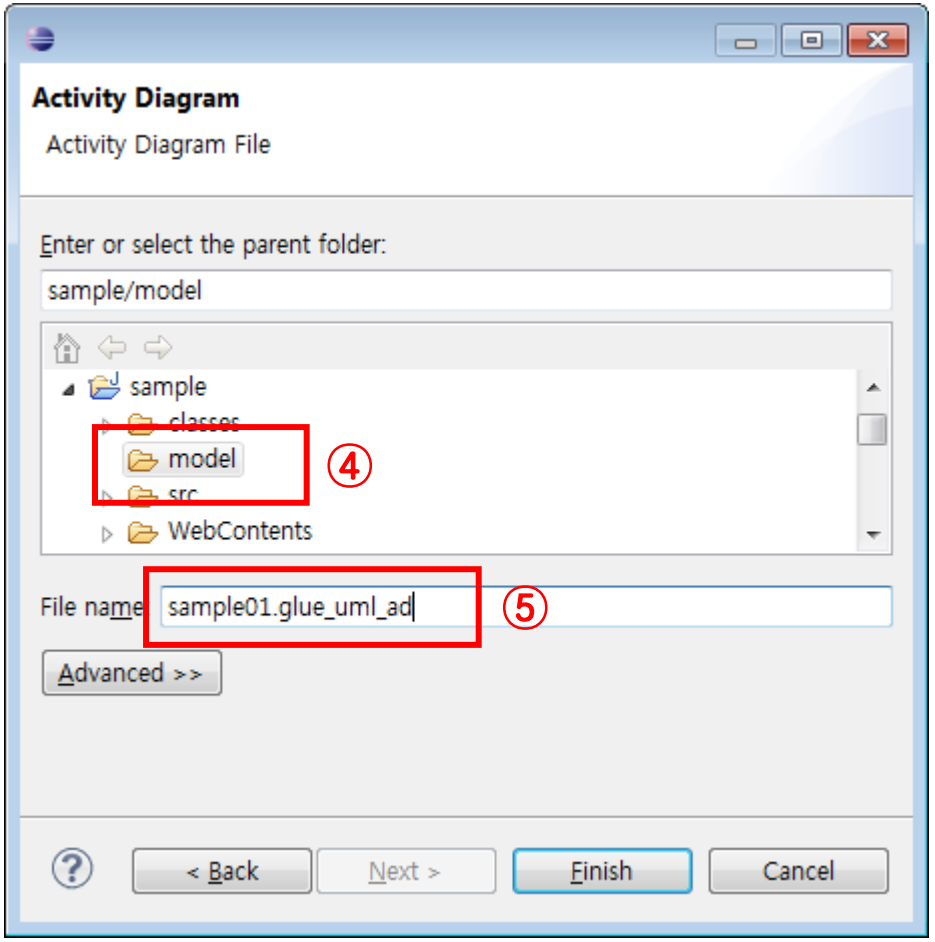
Glue Activity Diagram 기능

- System Level의 Business 설계
  - Activity Diagram을 통해 기능(Activity)와 기능들의 Flow를 정의
- Service 자동 생성
  - Glue F/W의 Business Control 에서 제어 되는 Service.xml이 생성
- Service 명세서 자동 생성
  - 설계된 Service 명세서를 자동으로 생성함.
    - System Level의 Use Case Diagram과 명세서, Class Diagram, Sequence Diagram, 기능 상세 정의서를 대체 할 수 있다.
- Service에 정의된 Custom Class 자동 생성
  - Service에 정의된 Custom Class를 Service 생성시 자동 생성함.
- Activity 정의 기능
  - Activity를 정의하고 개발자와 설계자간의 Communication 역할을 함.
- Reusable Activity 자동 Setting 기능
  - Preference에 등록하여 반복적인 작업을 줄여줌.
- Site 공통 Activity 정의 및 등록 기능
  - Glue 배포 Activity 뿐만이 아니라 Custom으로 작성된 Activity도 팔레트에 등록하여 사용 할 수 있음.

- 기능 3 : Glue Activity Diagram
  - Glue Activity Diagram 생성
    - wizard 실행.



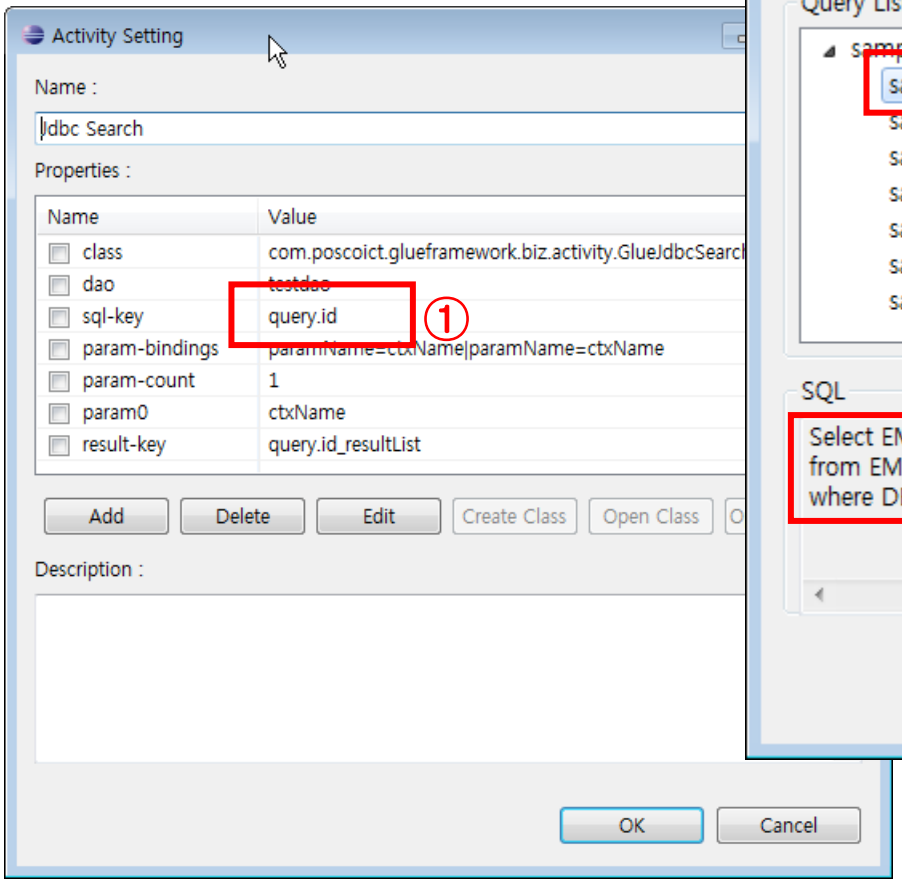
- 기능 3 : Glue Activity Diagram
  - Glue Activity Diagram 생성
    - wizard 실행
      - ④ 파일 위치 지정.
    - File name 수정
      - ⑤ name.glue_uml_ad
      - name은 프로그램ID 권장.
      - Glue Service 파일은 name-service.xml 으로 생성됨.

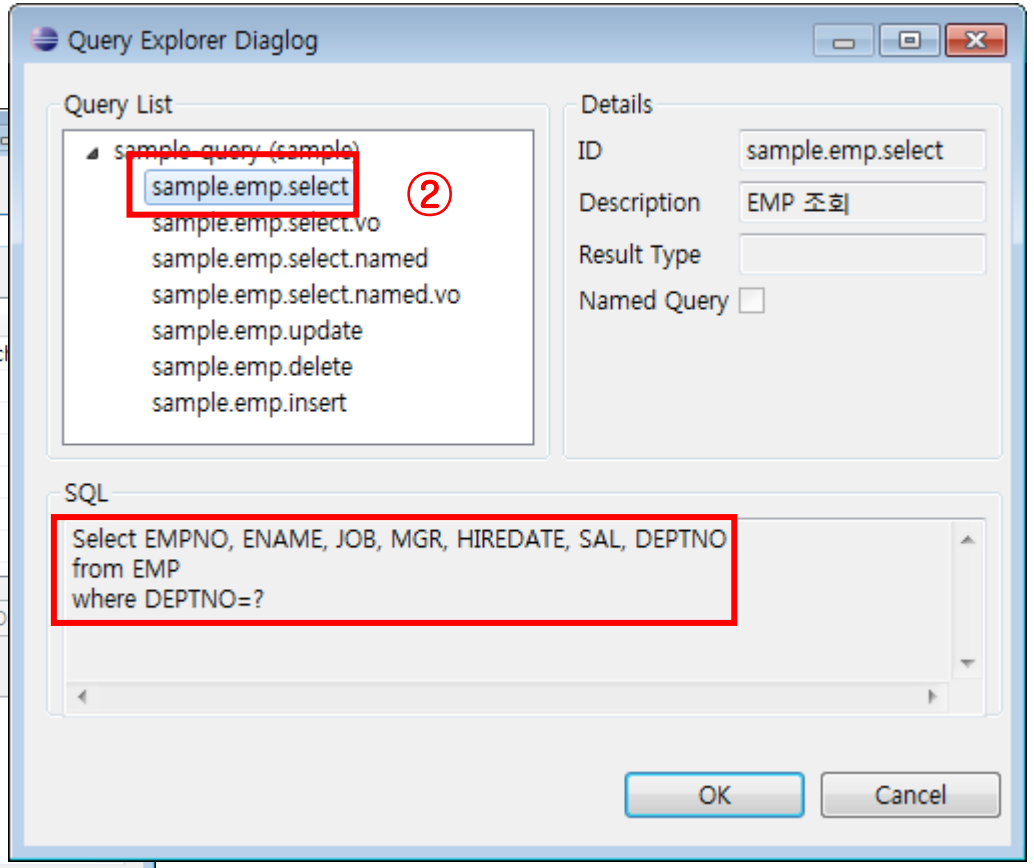


기능 3 : Glue Activity Diagram

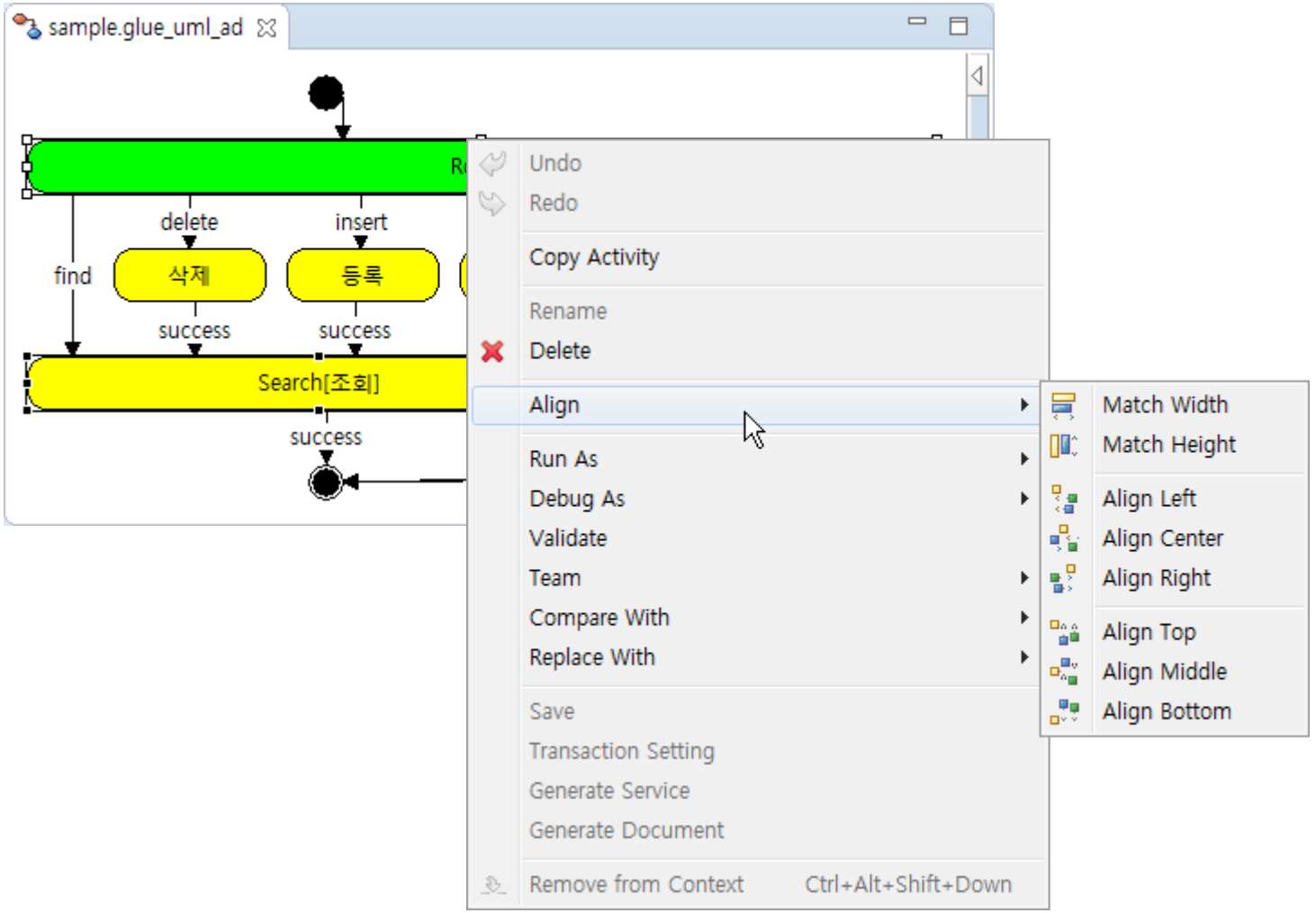
Query 선택 기능

- sql-key 의 value입력창 더블클릭
- Query Tree 에서 query 선택

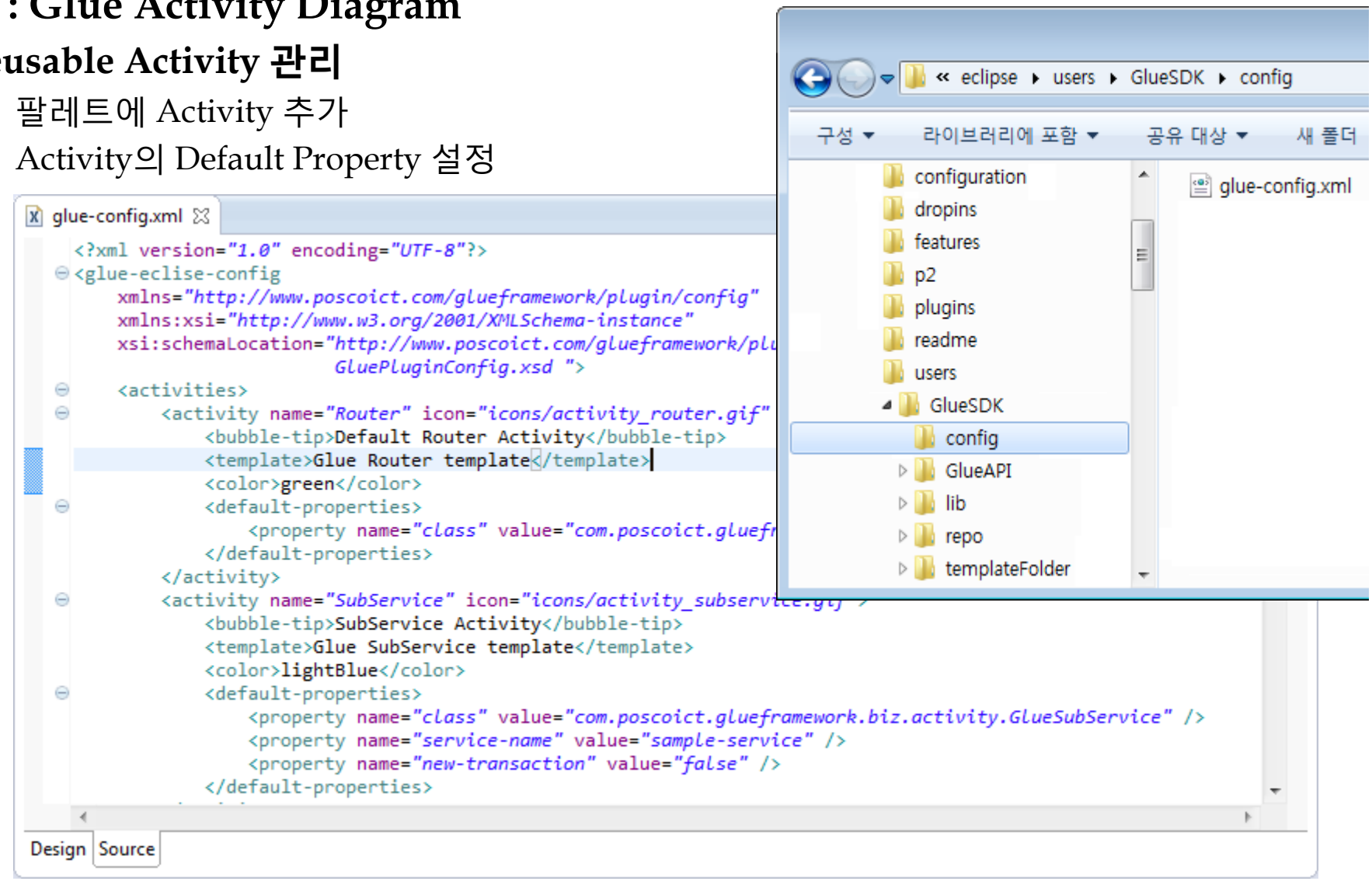




- 기능 3 : Glue Activity Diagram
  - Activity 정렬 기능



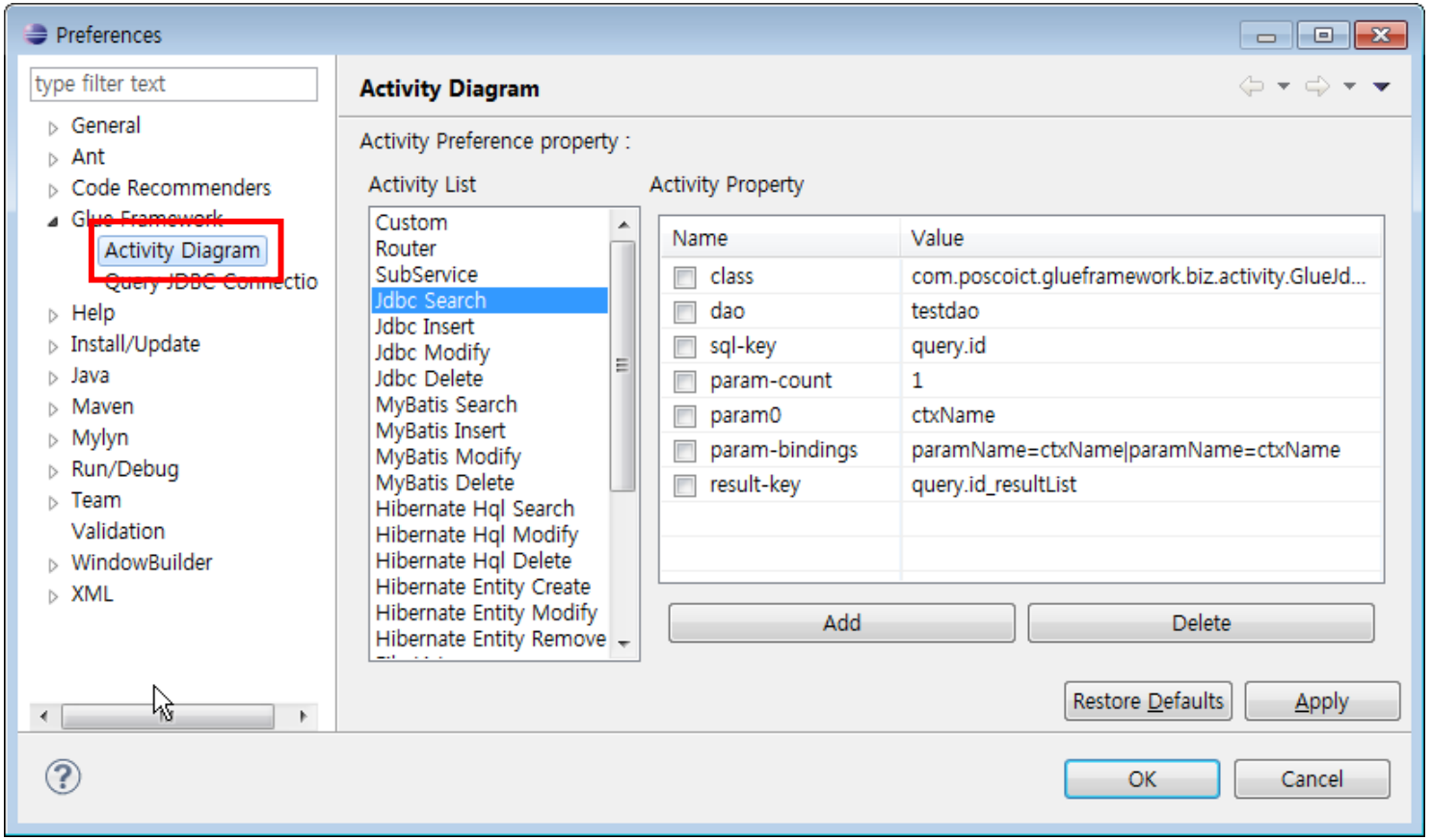
- 기능 3 : Glue Activity Diagram
  - Reusable Activity 관리
    - 팔레트에 Activity 추가
    - Activity의 Default Property 설정



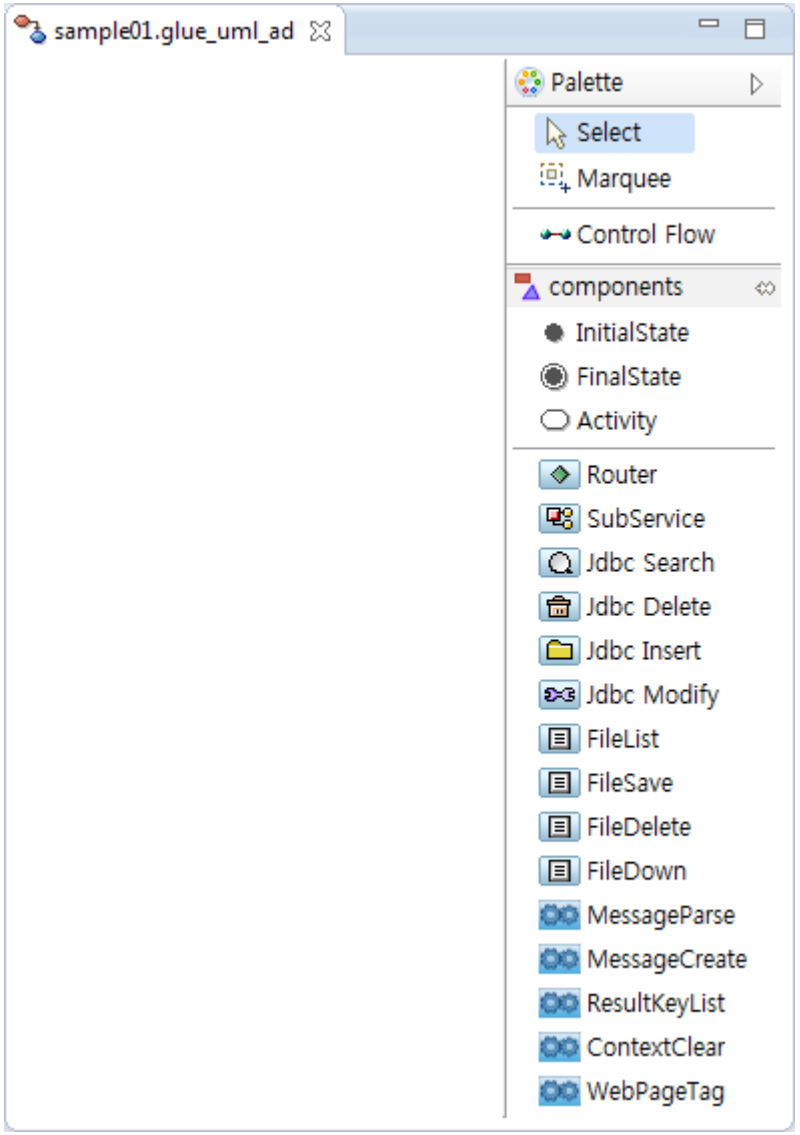
기능 3 : Glue Activity Diagram

Reusable Activity 관리

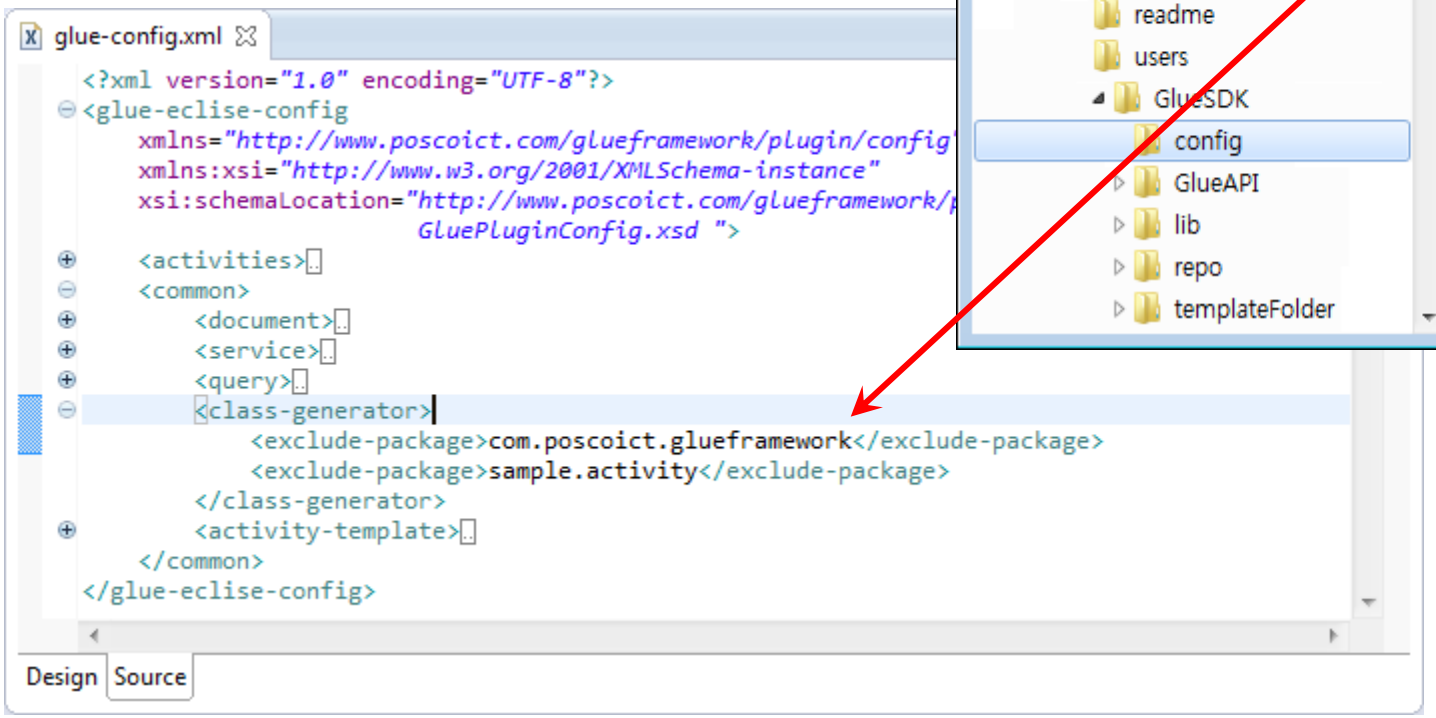
- 팔레트에 Activity 추가
- Activity의 Default Property 설정
- Activity 목록 및 Property 확인

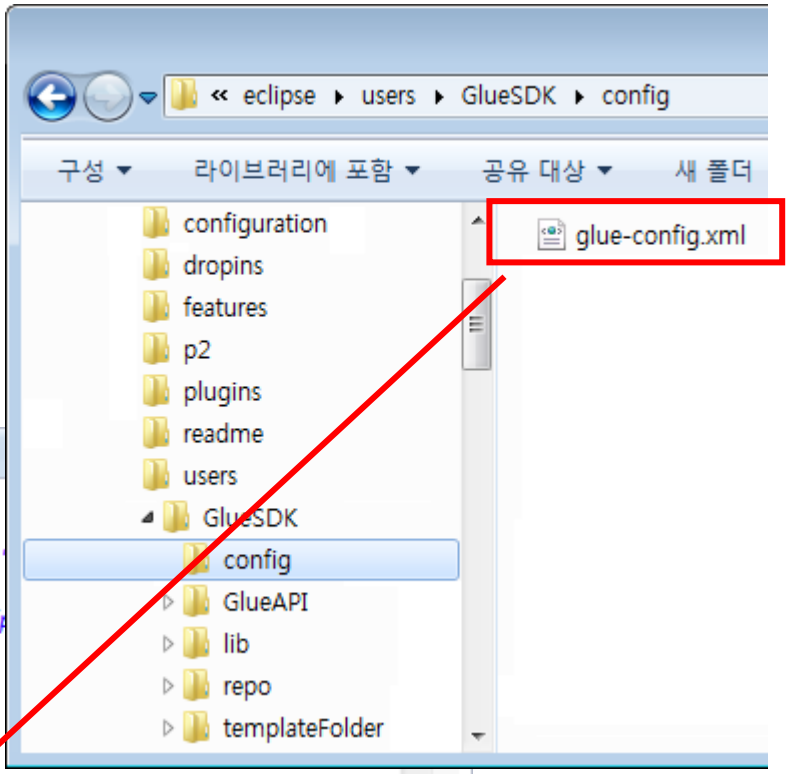


- **기능 3 : Glue Activity Diagram**
  - **Reusable Activity 관리**
    - 팔레트에 Activity 추가
    - Activity의 Default Property 설정
    - Activity 목록 및 Property 확인
    - 팔레트 확인

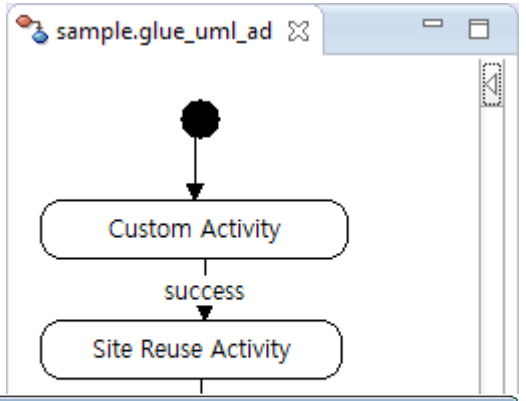


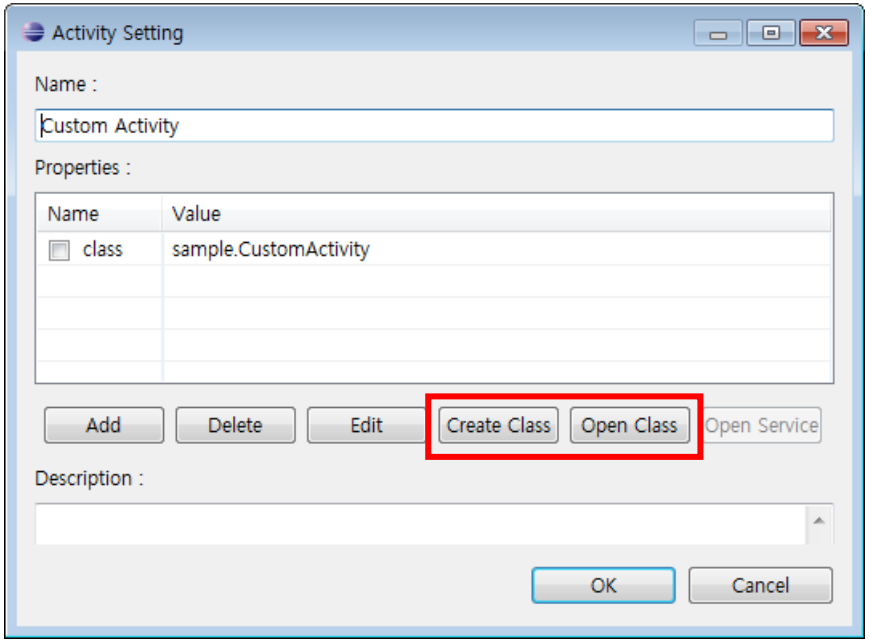
- 기능 3 : Glue Activity Diagram
  - Class 생성 기능
    - Sample Class 자동 생성
  - Class 생성 제외 기능
    - Reusable Class의 경우 생성 제외
    - glue-config.xml 설정

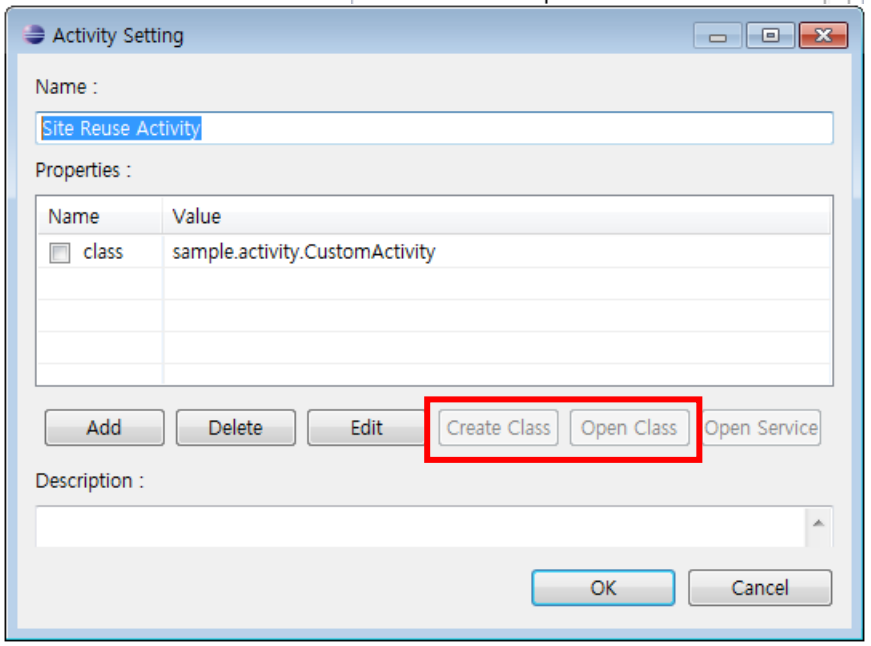




- 기능 3 : Glue Activity Diagram
  - Class 생성 기능
    - Sample Class 자동 생성
  - Class 생성 제외 기능
    - Reusable Class의 경우 생성 제외
    - glue-config.xml 설정
      - Create Class, Open Class 버튼의 활성화 여부





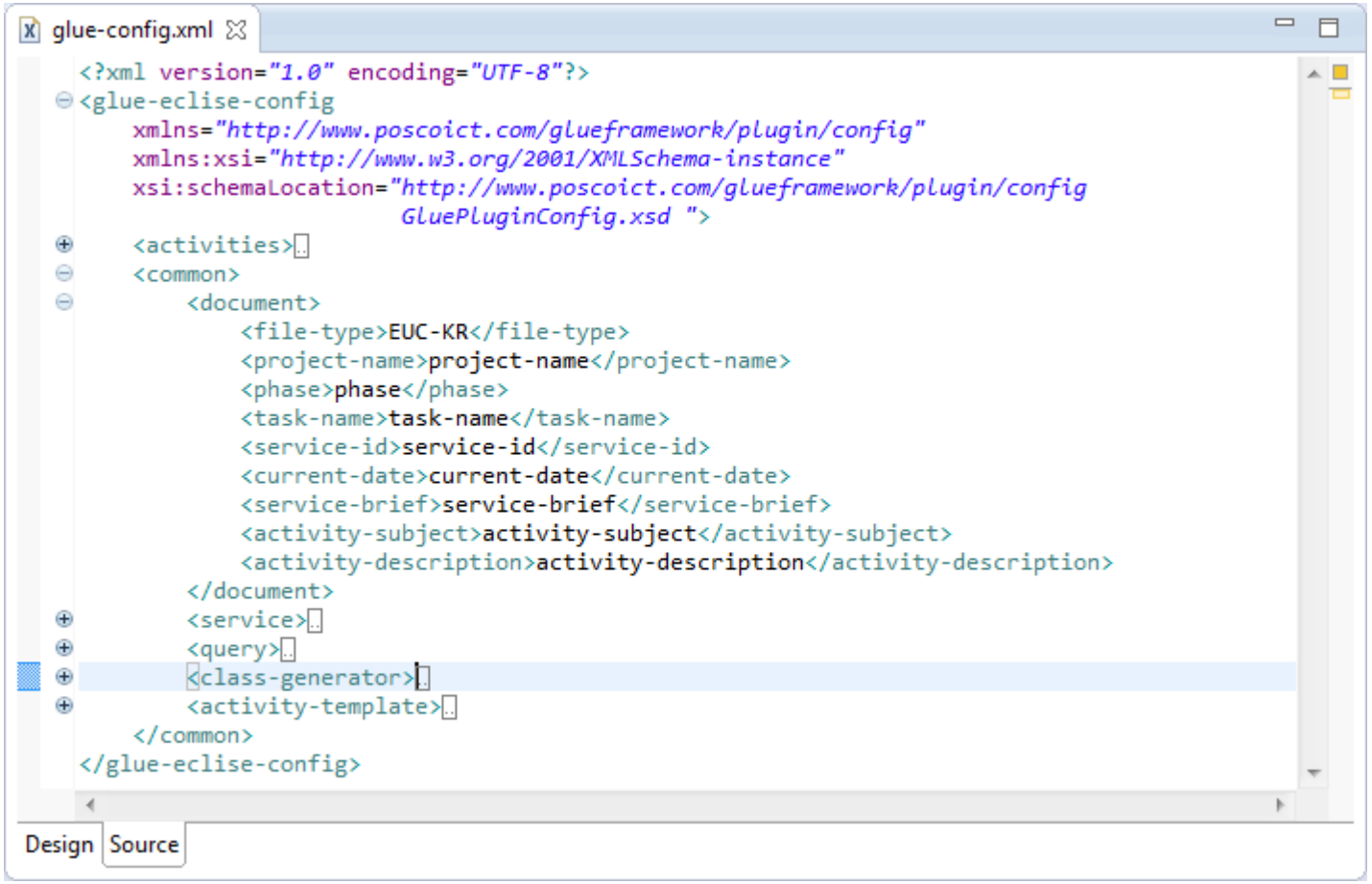


기능 3 : Glue Activity Diagram

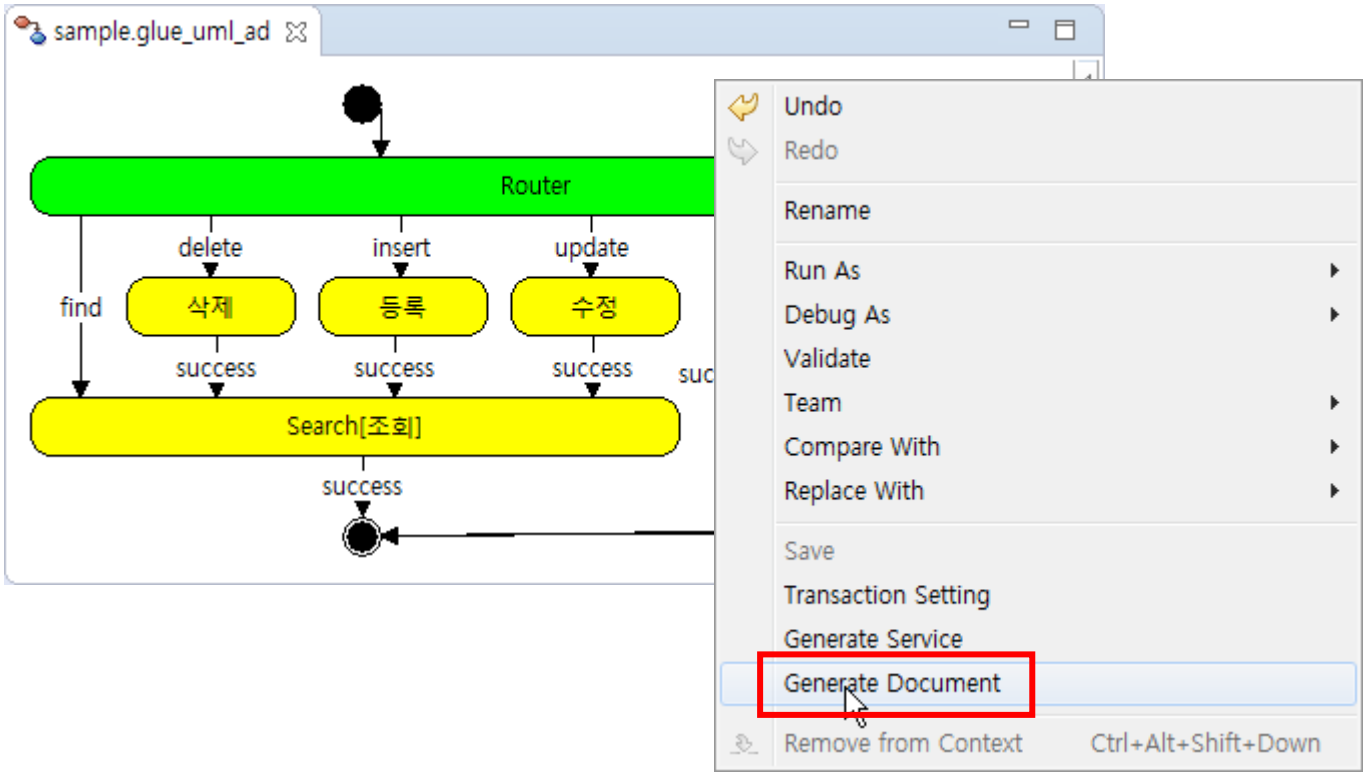
Service 명세서

환경 설정

- glue-config.xml 파일 : Document에 들어갈 제목을 정의 할 수 있음.



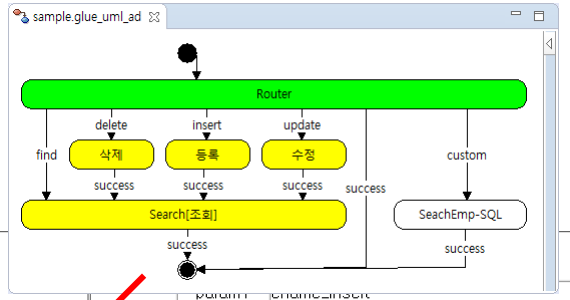
- 기능 3 : Glue Activity Diagram
  - Service 명세서
    - 생성 방법



기능 3 : Glue Activity Diagram

Service 명세서

- Activity Diagram은 화면을 Capture 하여 넣어야 함.



Service Specification

project-name	
phase	
task-name	
service-id	sample-service
current-date	2015-02-11

Service Name : sample-service

1. service-brief

GlueSDK에 포함된 예제 화면을 위한 Activity Diagram 입니다.
sample-service는 사원정보를 관리하는 서비스입니다.

2. Activity Diagram

Please put the Activity Diagram Image!

3. activity-subject

3.1 Router

Properties:

name	value
class	com.poscoict.glueframework.biz.activity.GlueDefaultRouter

activity-description:

화면에서 발생되는 Event 에 따라 분기하는 Activity입니다.
GlueContext 에는 find, delete, insert, update 중 한개의 정보가 있습니다.

3.2 등록

Properties:

name	value
class	com.poscoict.glueframework.biz.activity.GlueJdbcInsert
sql-key	[sample.emp.insert] insert into emp(empno, ename, sal, deptno) values(?, ?, ?, ?)
chk-name	chk_insert
dao	test-dao
param0	empno_insert
param1	ename_insert
param2	sal_insert
param3	deptno_insert
param-count	4
result-key	insertCnt

activity-description:

사원정보를 추가하는 Activity입니다.
화면에서 입력한 정보(사원번호, 사원명, 급여, 부서번호)를 이용해 EMP Table에 record를 추가합니다.

3.3 삭제

Properties:

name	value
class	com.poscoict.glueframework.biz.activity.GlueJdbcDelete
sql-key	[sample.emp.delete] delete from emp where empno=?
chk-name	chk
dao	test-dao
param0	EMPNO
param-count	1

activity-description:

사원정보를 삭제하는 Activity입니다.
result-key 를 설정하지 않았으므로, 삭제결과는 '{sql-key}_deleteCnt' 가 사용됩니다.

3.4 수정

Properties:

name	value

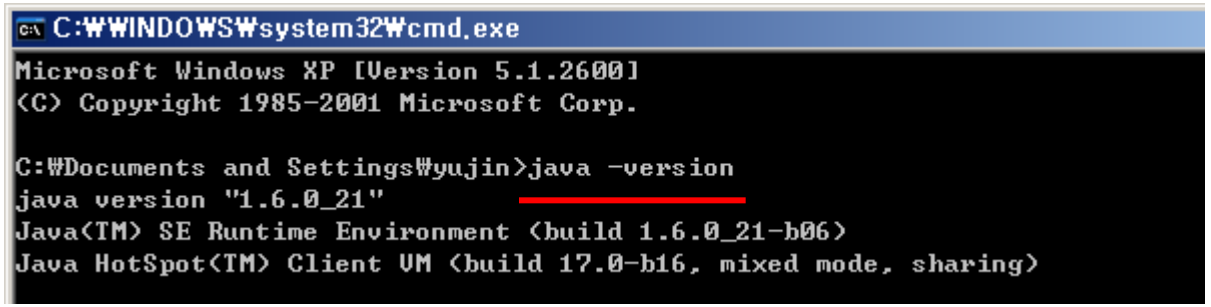
- **기능 4 : Glue Meta Data File Type 지정**
  - **File Type 지정 가능 File**
    - Service, Query, Document File
  - **File Type 지정 방법**
    - glue-config.xml 에 설정

glue-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<glue-eclise-config
    xmlns="http://www.poscoict.com/glueframework/plugin/config"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.poscoict.com/glueframework/plugin/config
                        GluePluginConfig.xsd ">
    <activities>..</activities>
    <common>
        <document>
            <file-type>EUC-KR</file-type>
            <project-name>project-name</project-name>
            <phase>phase</phase>
            <task-name>task-name</task-name>
            <service-id>service-id</service-id>
            <current-date>current-date</current-date>
            <service-brief>service-brief</service-brief>
            <activity-subject>activity-subject</activity-subject>
            <activity-description>activity-description</activity-description>
        </document>
        <service>
            <file-type>UTF-8</file-type>
        </service>
        <query>
            <file-type>UTF-8</file-type>
        </query>
```

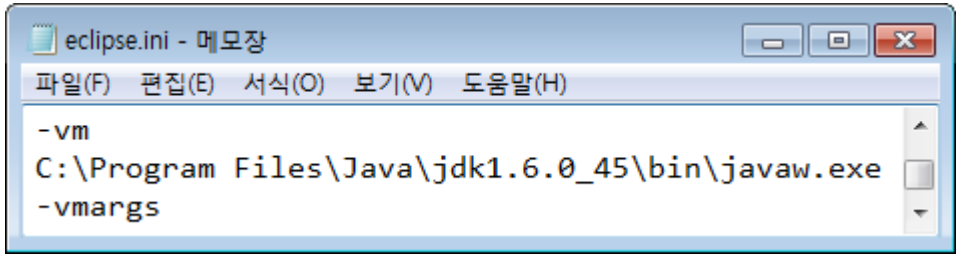
II. Glue Framework 환경구성

- Java
  - JAVA 버전 확인



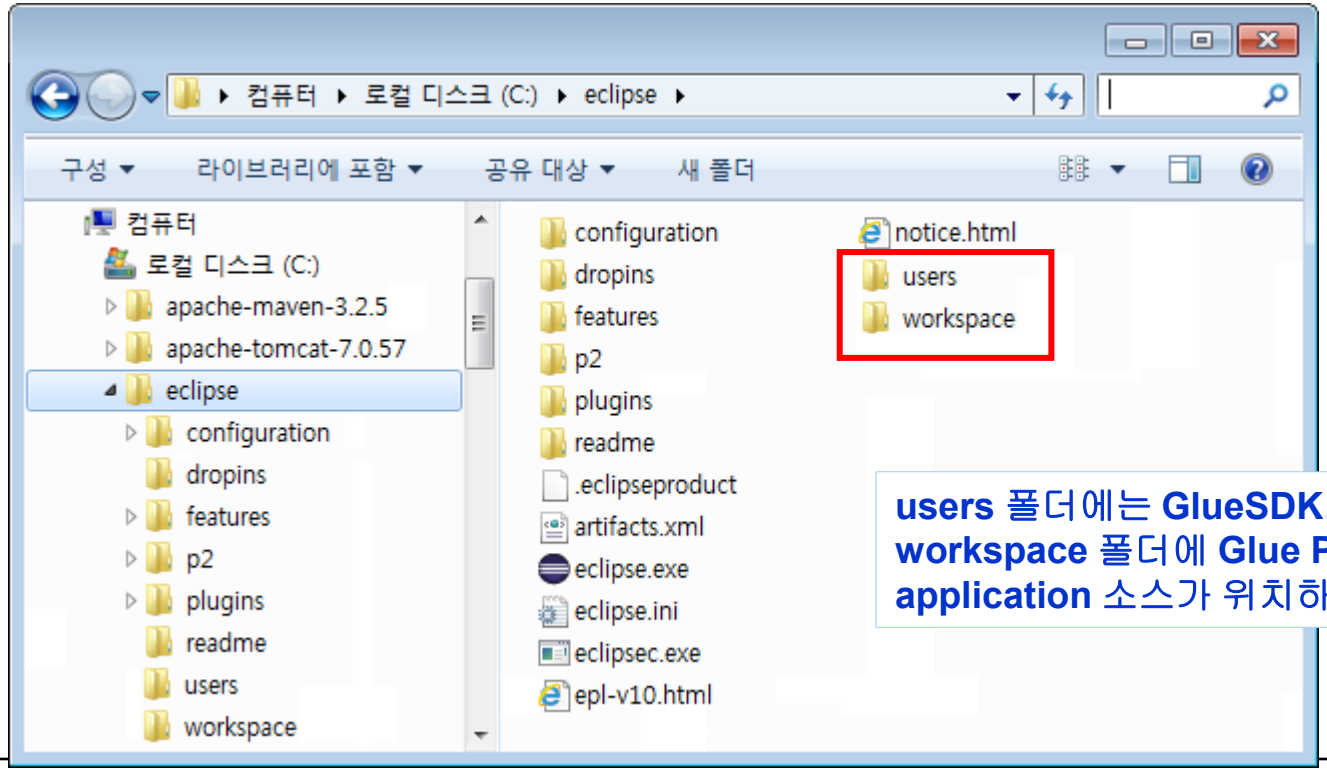
    - Java SE 6 이상이어야 함.
      - Glue Framework 은 Java SE 6에서 개발되었음.
      - Eclipse 4.x에서 Java SE 6 이상을 요구함.
      - Tomcat 7.x 에서 Java SE 1.6 이상을 요구함.
    - Eclipse
      - eclipse.ini 에서 –vm 옵션으로 JVM 지정가능
    - Tomcat
      - bin/setclasspath.bat 에서 JAVA_HOME 지정 가능

- **Eclipse 설치**
  - **Eclipse 설치 - eclipse-java-xxxx-xxxxx.zip 압축해제**
    - ECLIPSE_HOME : Eclipse 설치 위치
      - 환경 파일 수정
        - 파일 위치 : %ECLIPSE_HOME% /eclipse.ini
        - -vm 옵션으로 JVM 지정가능
        - -vmargs 위에 추가



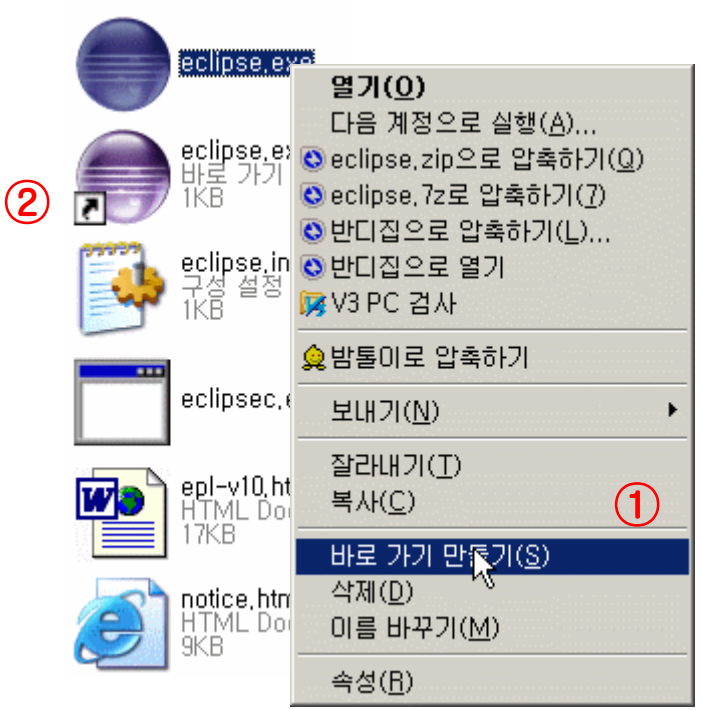
      - Eclipse 실행 여부 확인 후 종료

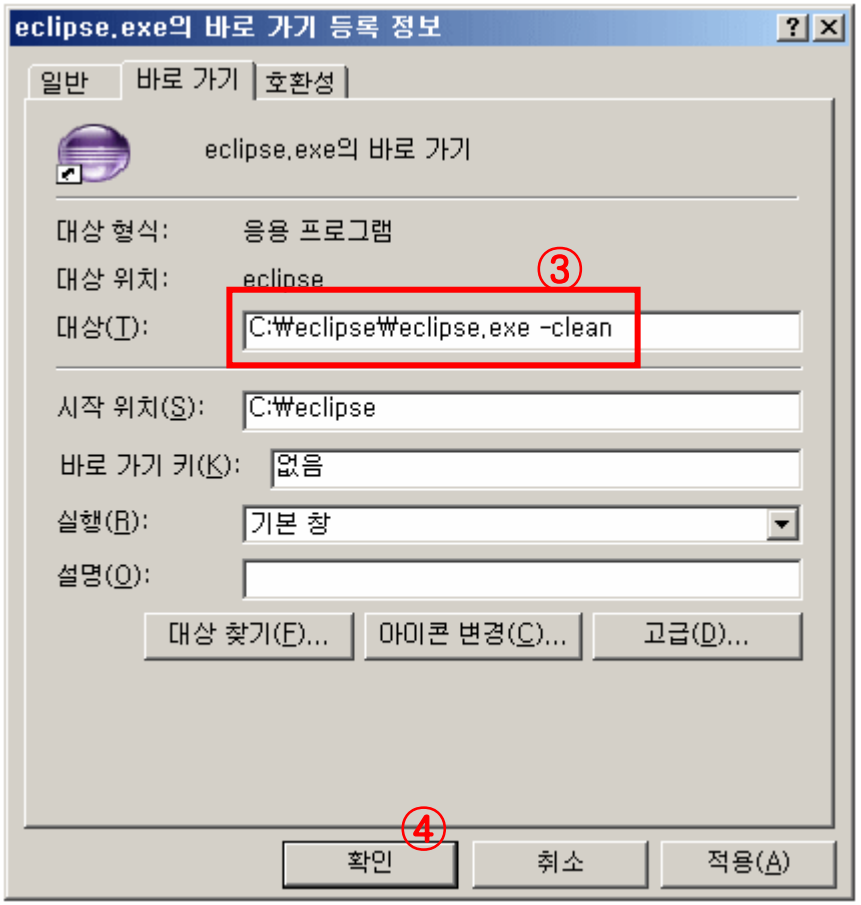
- Eclipse 설치
  - Eclipse 설치 - eclipse-java-xxxx-xxxxx.zip 압축 해제
    - ECLIPSE_HOME : Eclipse 설치 위치
    - users 폴더 생성
      - 위치 : %ECLIPSE_HOME%/users
    - workspace 폴더 생성
      - 위치 : %ECLIPSE_HOME%/workspace



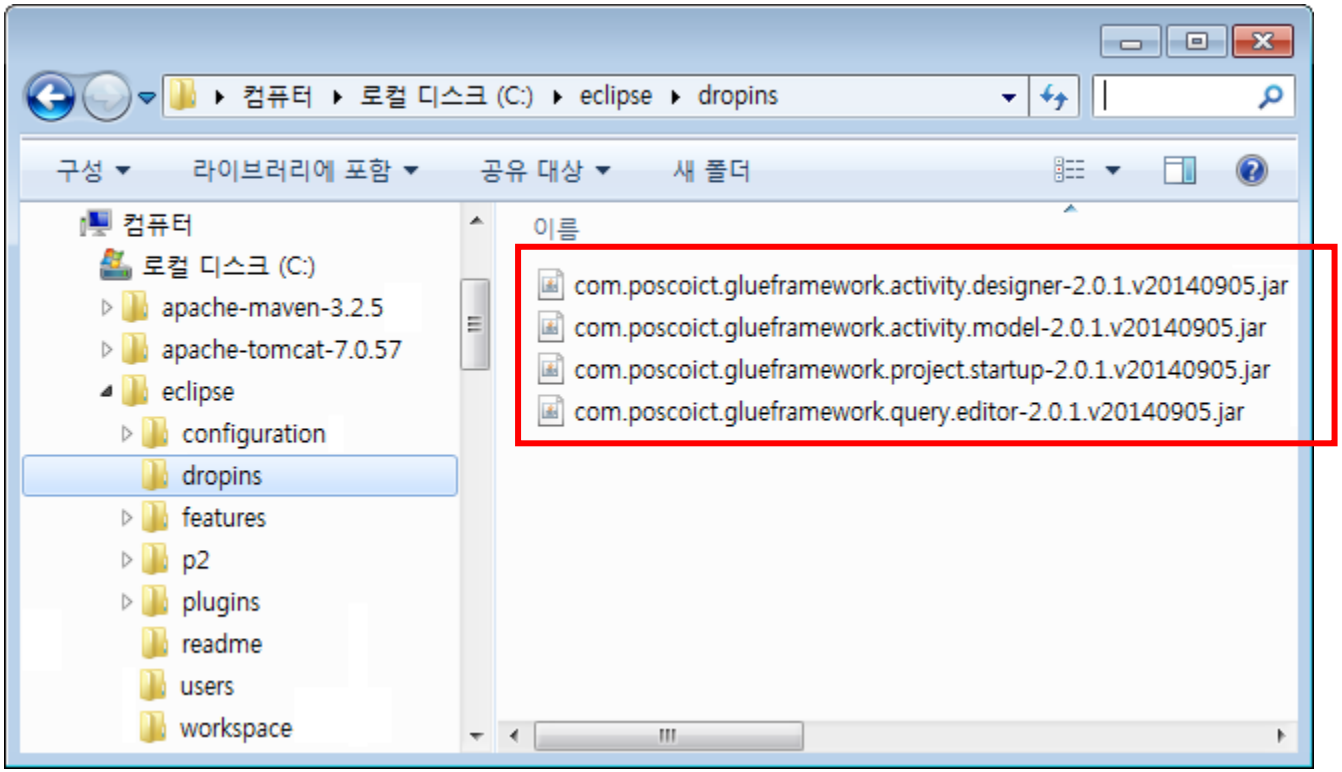
users 폴더에는 **GlueSDK**가 설치된다.
workspace 폴더에 **Glue Project**가 생성되고,
application 소스가 위치하게 된다.

- Eclipse 설치
  - Eclipse 설치 - eclipse-java-xxxx-xxxxx.zip 압축 해제
    - ECLIPSE_HOME : Eclipse 설치 위치
    - users 폴더 생성
    - workspace 폴더 생성
    - 바로 가기 만들어서 –clean 옵션주기

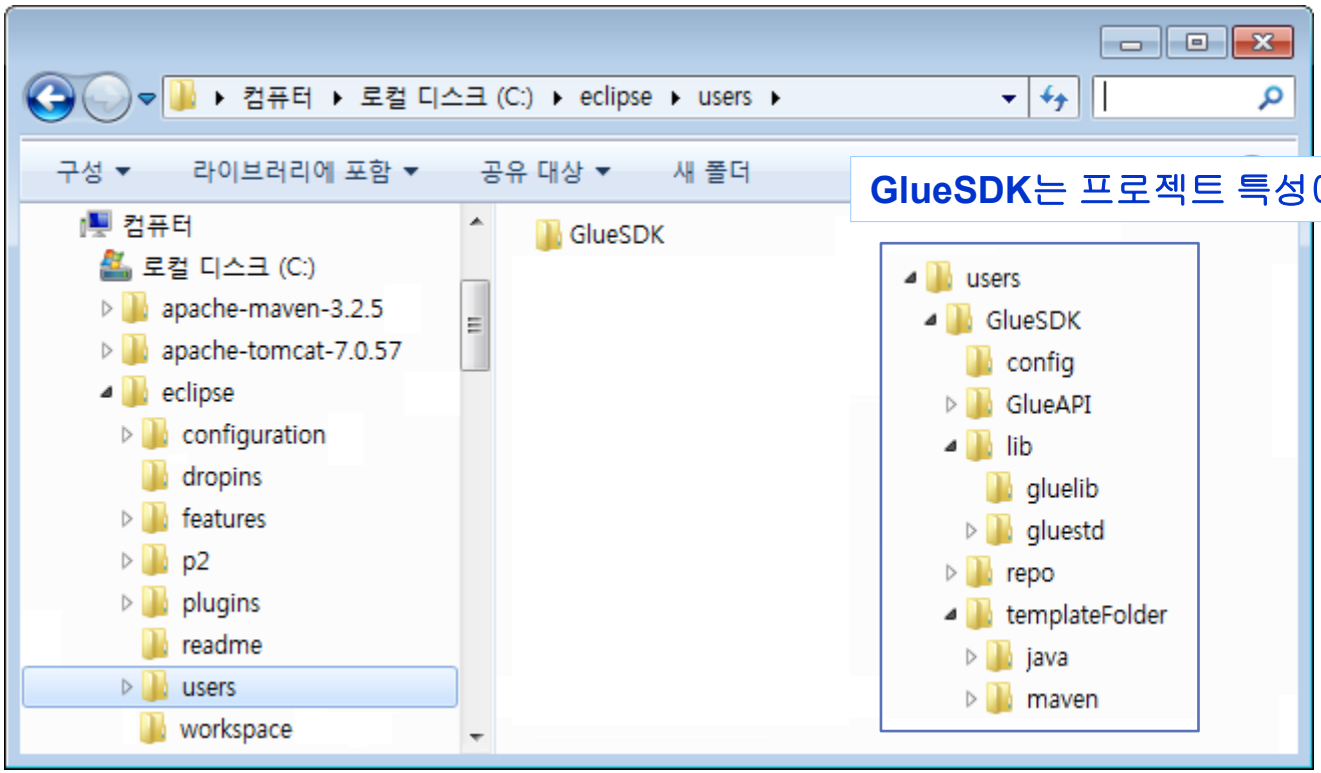




- Eclipse 설치
  - Eclipse 설치
  - **plugin & GlueSDK 설치**
    - plugin 설치 : %ECLIPSE_HOME%/dropins 에 glue plugin 4개 복사

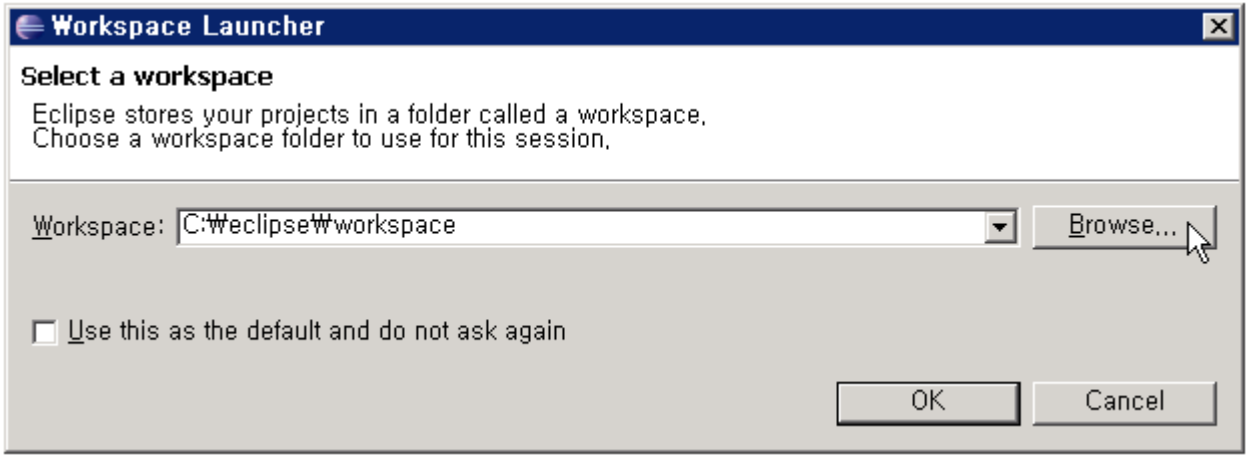


- Eclipse 설치
  - Eclipse 설치
  - plugin & GlueSDK 설치
    - plugin 설치
    - GlueSDK 설치 : %ECLIPSE_HOME%/users 에 GlueSDK 복사

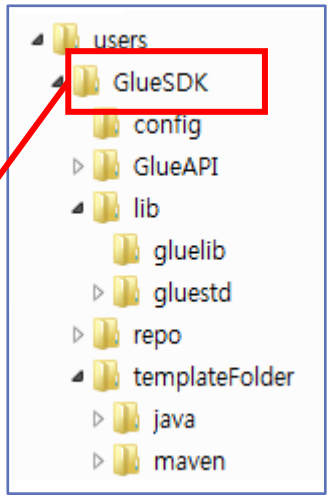


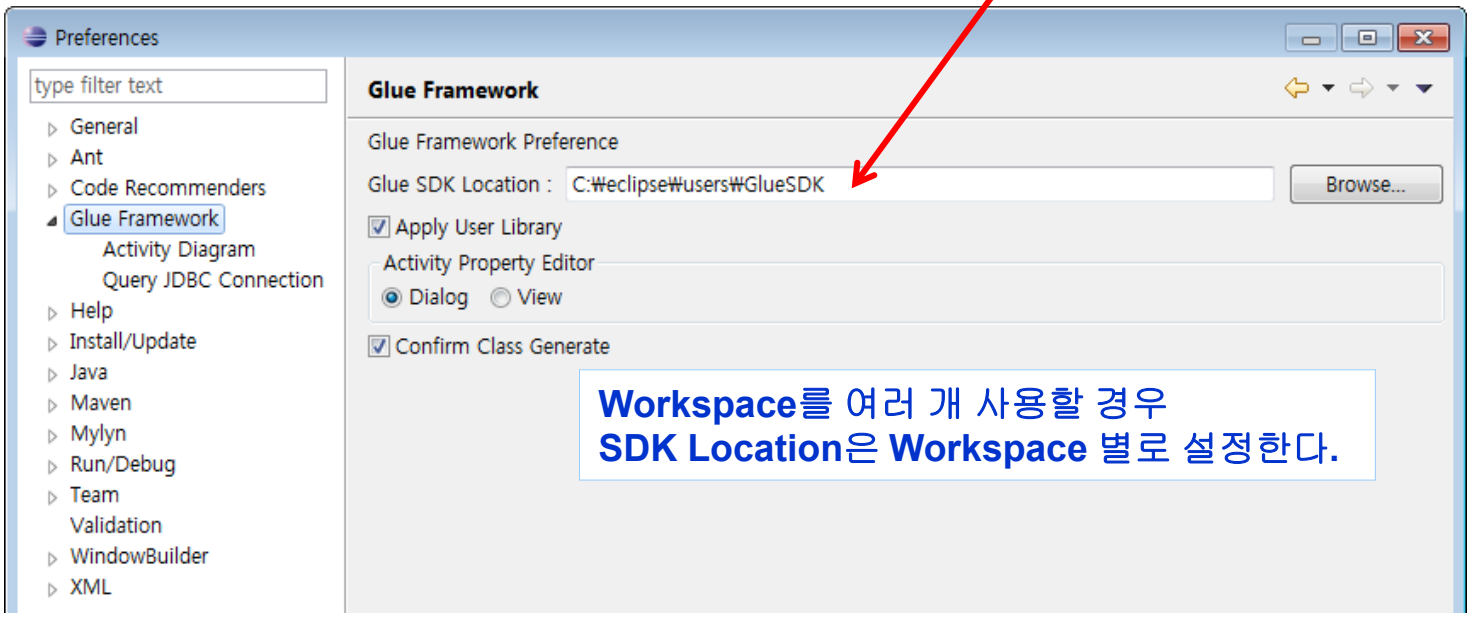
GlueSDK는 프로젝트 특성에 맞게 관리한다.

- Eclipse 설치
  - Eclipse 설치
  - plugin & GlueSDK 설치
    - plugin 설치
    - GlueSDK 설치
    - GlueSDK 지정 : Workspace 지정(최초 1회)
      - Eclipse 실행 → Workspace 지정
        - %ECLIPSE_HOME%/workspace

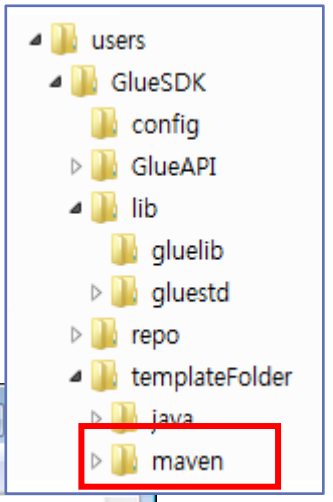


- Eclipse 설치
  - Eclipse 설치
  - plugin & GlueSDK 설치
    - plugin 설치
    - GlueSDK 설치
    - GlueSDK 지정
      - Eclipse 실행 → Window → Preferences → Glue Framework
        - SDK Location 의 'Browse...' 클릭
        - %ECLIPSE_HOME%\users\GlueSDK 로 지정





- Eclipse 설치
  - Eclipse 설치
  - plugin & GlueSDK 설치
  - **GlueSDK 수정**
    - pom.xml 수정 : %GlueSDK% / templateFolder / maven / pom.xml
      - repo의 위치(<url>)을 수정



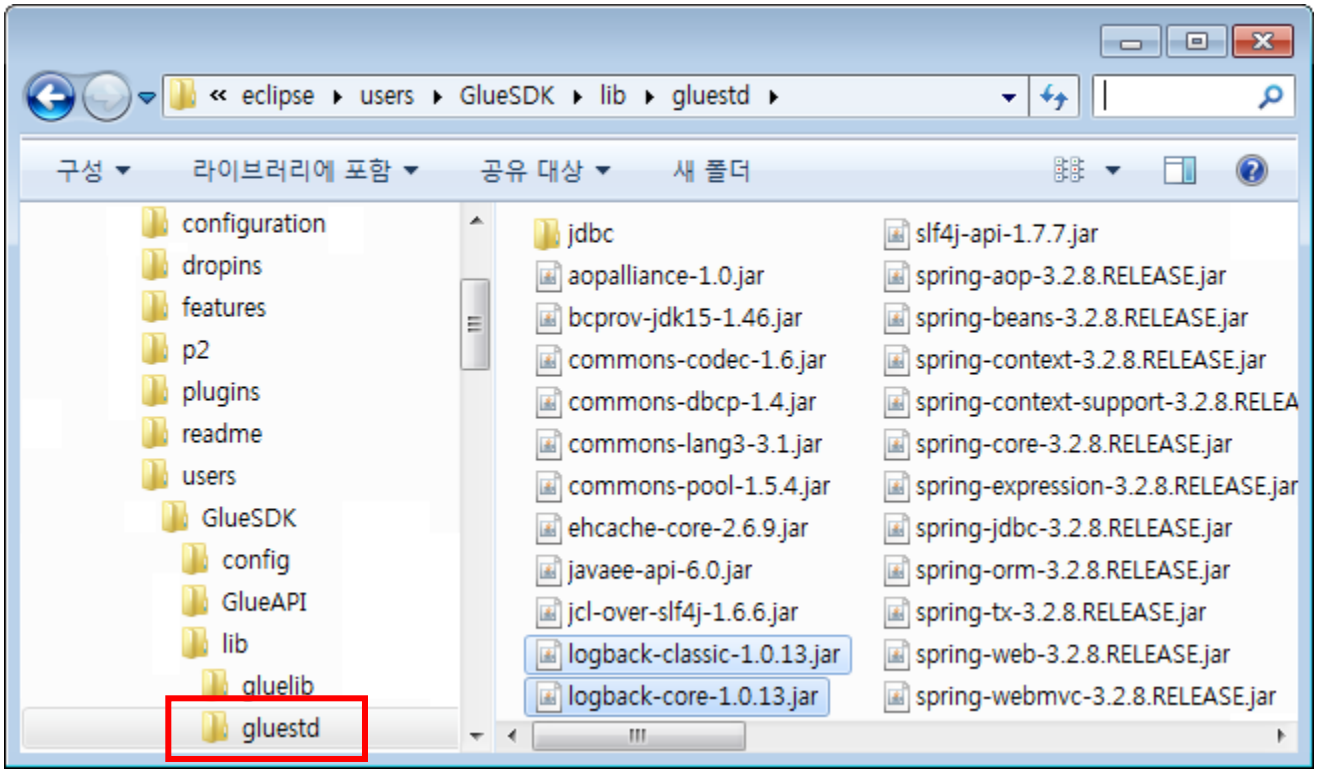
공용 **repository**가 있다면 **<repositories>**는 삭제하세요.

- Eclipse 설치
  - Eclipse 설치
  - plugin & GlueSDK 설치
  - GlueSDK 수정
    - pom.xml 수정
    - gluestd 에 라이브러리 추가

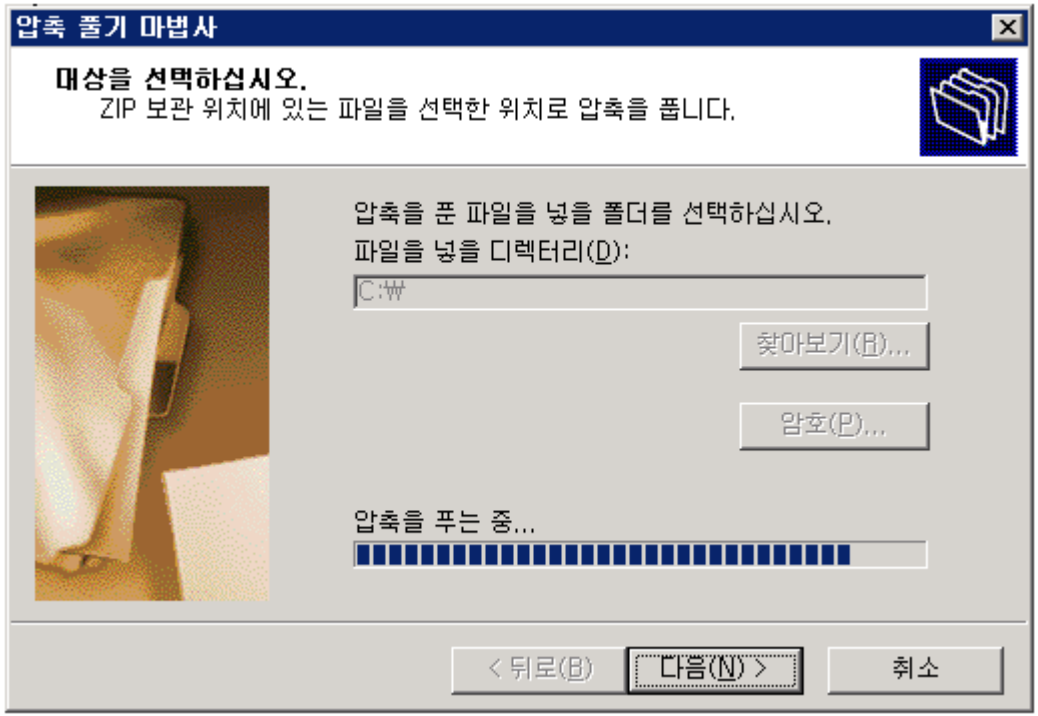
Logback을 사용하기 위해서는
다음 **2**개 **Libraray**는 **GlueSDK/lib/gluestd** 에 추가한다.

-- logback-classic-{version}.jar
-- logback-core-{version}.jar

다운로드 : **http://logback.qos.ch**



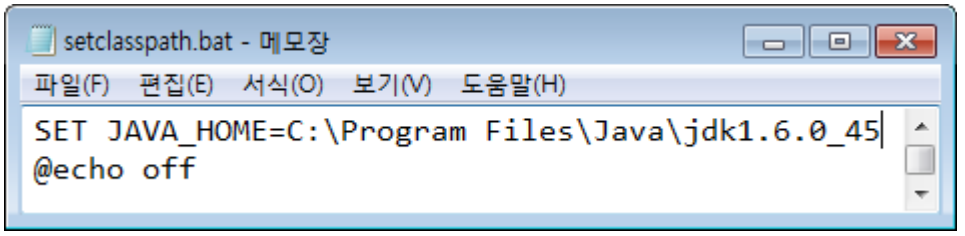
- Maven 설치
  - Maven 설치 - apache-maven-3.2.x-bin.zip 압축 해제
    - MAVEN_HOME : Maven 설치위치



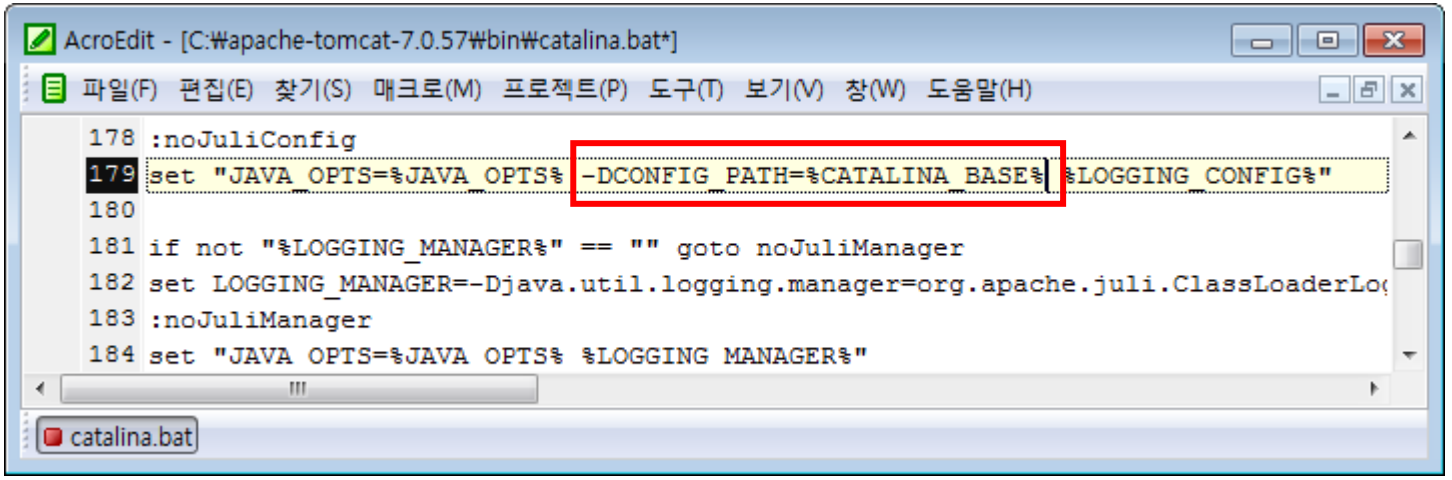
Tomcat 설치

Tomcat 설치 - apache-tomcat-7.0.x.zip 압축 해제

- CATALINA_HOME : Tomcat 설치위치
- 실행 파일 수정
  - %CATALINA_HOME%/bin/setclasspath.bat
    - JAVA_HOME 을 맨 위에 추가



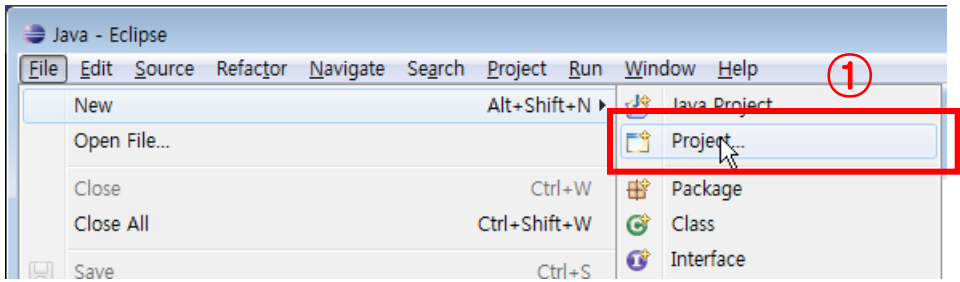
  - %CATALINA_HOME%/bin/catalina.bat

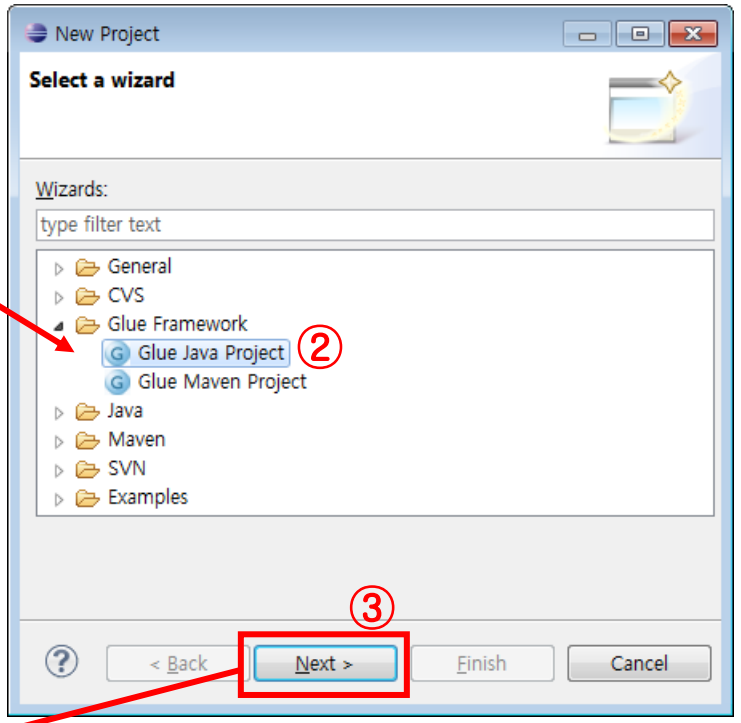


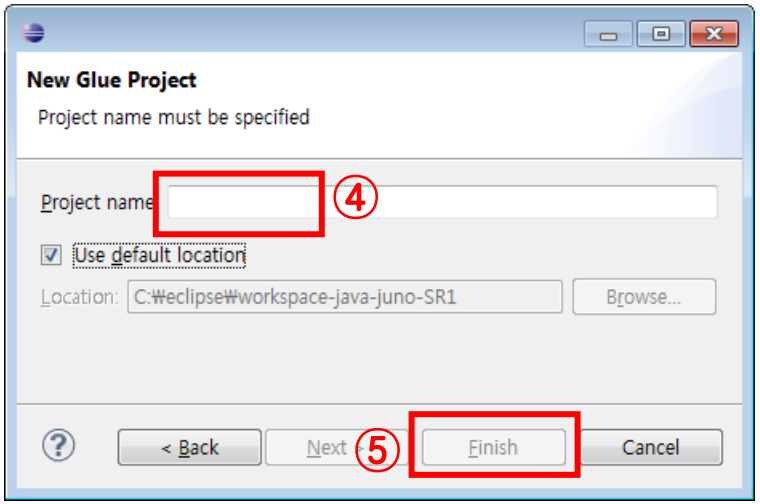
- **Tomcat 설치**
  - Tomcat 설치
  - **Tomcat 확인**
    - %CATALINA_HOME%/bin/startup.bat
    - http://127.0.0.1:8080
    - %CATALINA_HOME%/webapps/ROOT 에 check.jsp 추가
      - http://127.0.0.1:8080/check.jsp

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8">
</head>
<body>
<form method=post>
<input type=text name="key" value="CONFIG_PATH">
<input type=submit value="check">
</form><%
  if(request.getParameter("key")!=null){
    out.println(System.getProperty(request.getParameter("key")));
  }
%>
<hr>
</body>
</html>
```

- Glue Project 생성
  - File > New > Project 선택
  - Wizard에서 Glue Framework > Glue Project 선택 > Next 버튼 클릭
  - Project name 입력
  - Finish



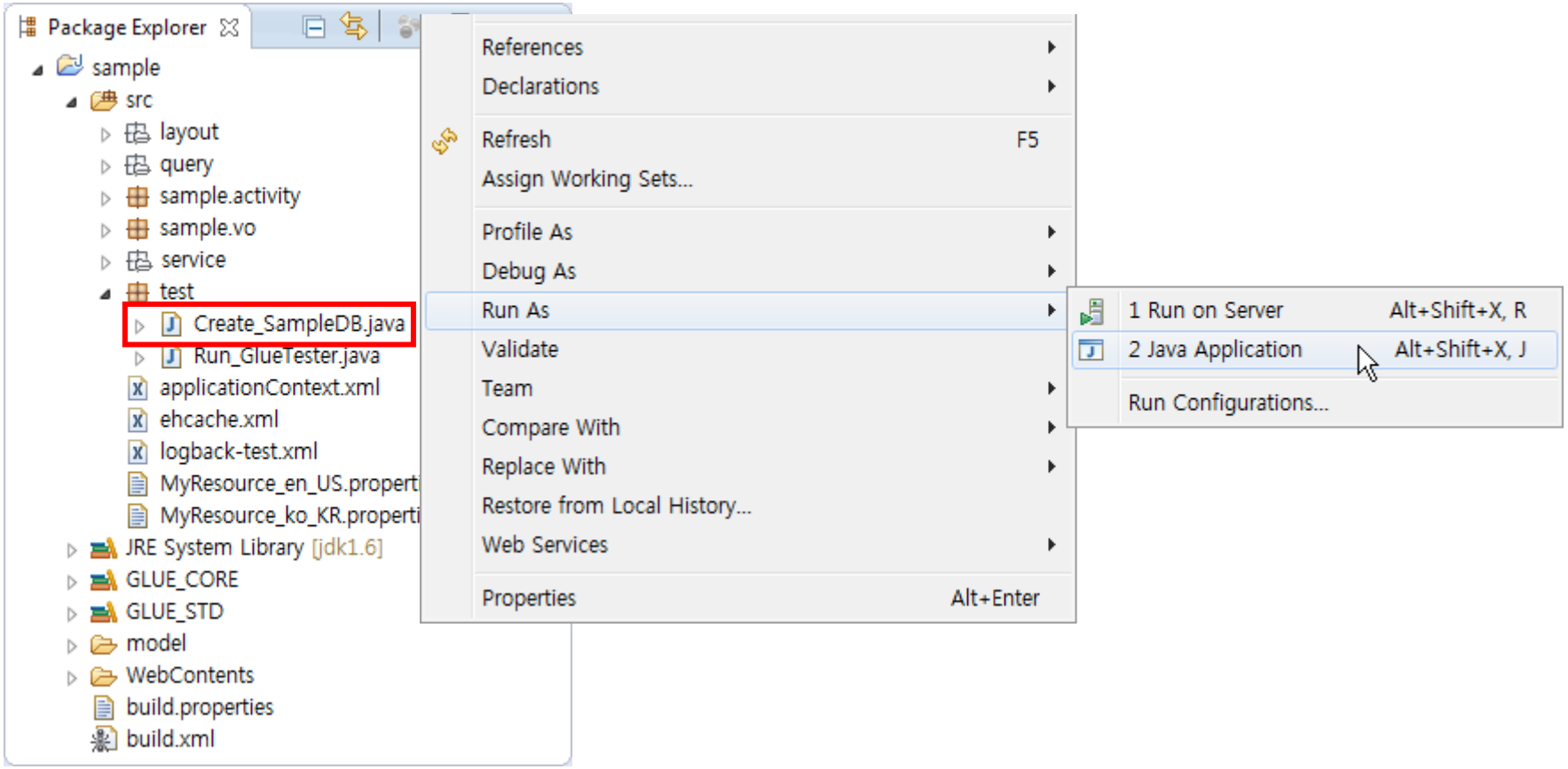




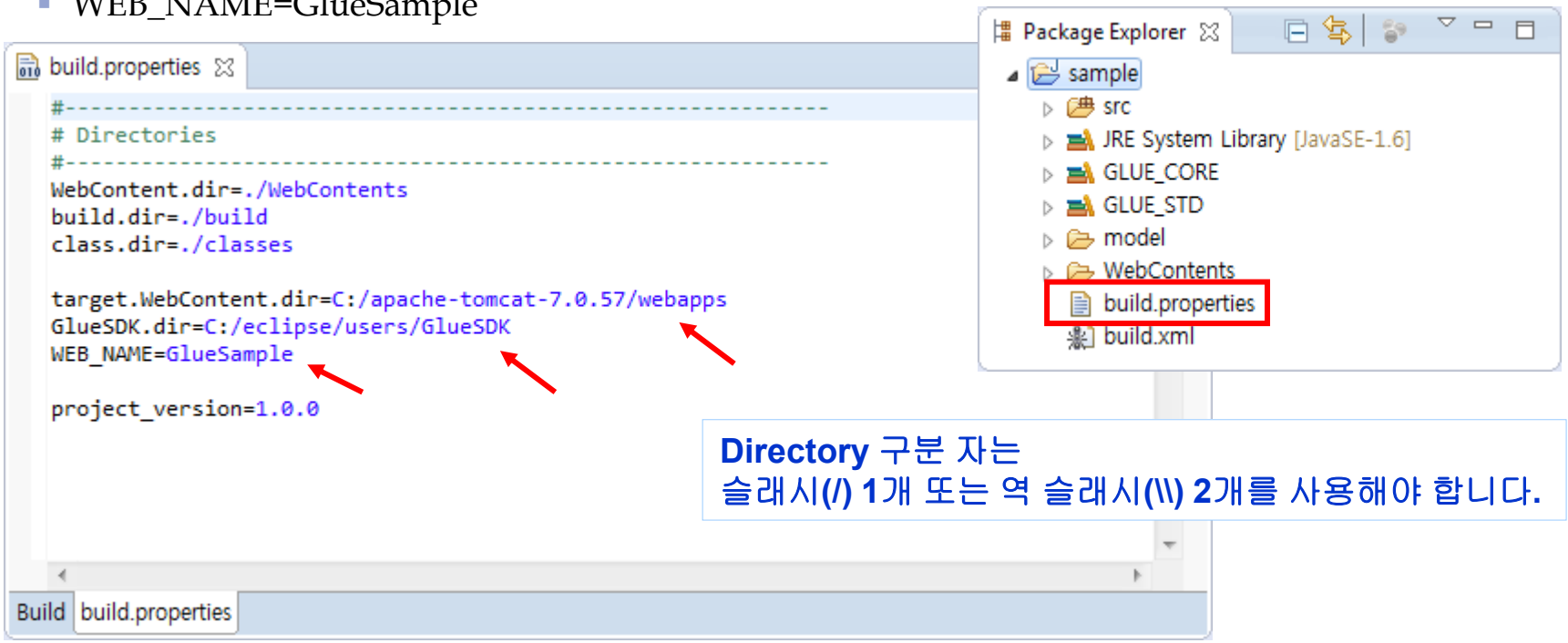
2. Sample Application [실습]

- 실습#1
  - Glue Java Project 유형으로 Project 생성
    - sample
  - Glue Maven Project 유형으로 Project 생성
    - sample-maven
    - sample-scheduler

- 실습#1-1
  - DB 생성
    - sample.db(파일DB)가 생성되며, %CATALINA_HOME% 으로 복사



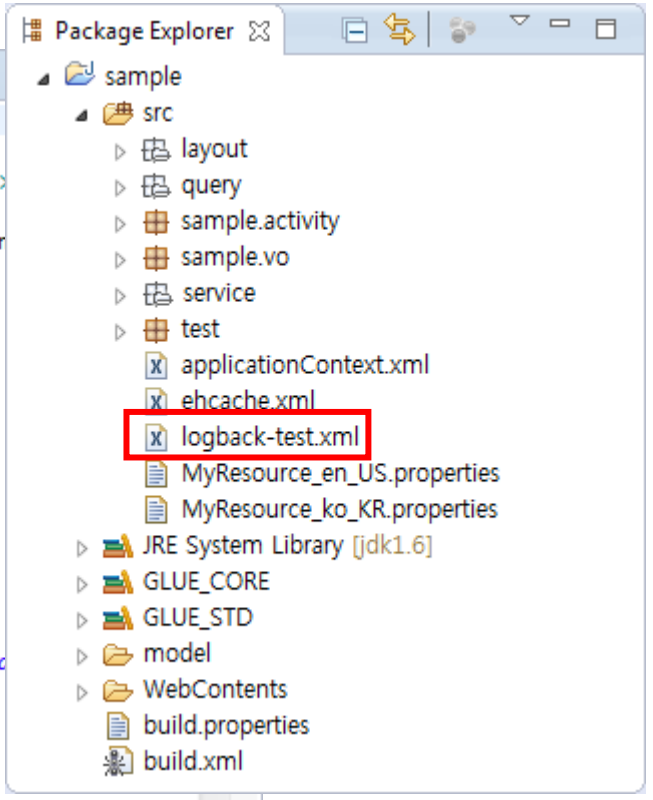
- 실습#1-2
  - build.properties 수정
    - Deploy 위치 지정
      - target.WebContent.dir=%CATALINA_HOME%/webapps
    - GlueSDK 위치 지정
      - GlueSDK.dir= %ECLIPSE_HOME%/users/GlueSDK
    - Web Application 이름 지정
      - WEB_NAME=GlueSample



- 실습#1-3
  - logback-test.xml 수정
    - glue log 파일 위치 지정 : FILE appender의 <file> 요소

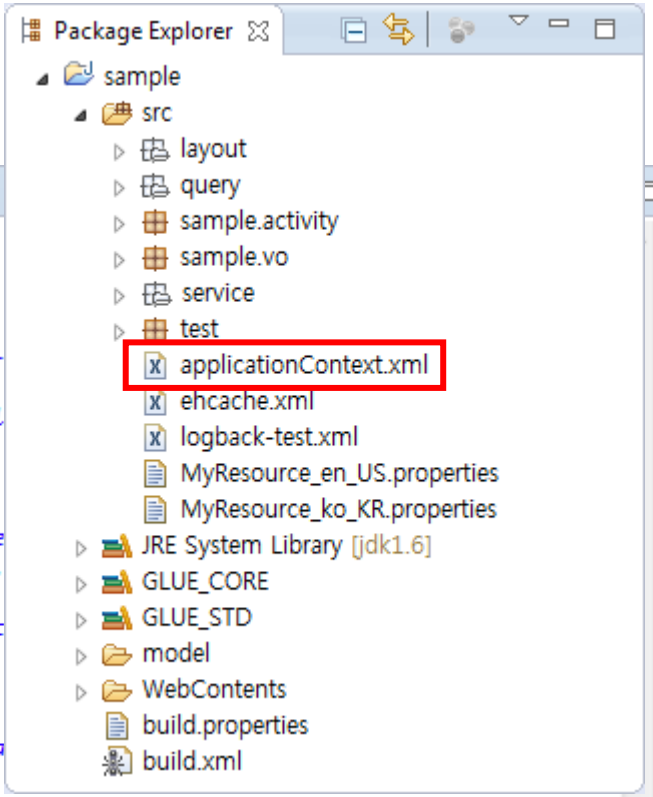
build.properties | logback-test.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration debug="true">
    <appender name="CONSOLE" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger - %msg%n
        </encoder>
    </appender>
    <appender name="FILE" class="ch.qos.logback.core.FileAppender">
        <file>glue_app_debug_log.log</file>
        <append>true</append>
        <encoder>
            <pattern>%d [%thread] %-5level %logger - %msg%n</pattern>
        </encoder>
    </appender>

    <logger name="com.poscoict.glueframework">
        <level value="debug" />
    </logger>
    <logger name="com.poscoict.glueframework.util.license.GlueLicenseMana
        <level value="trace" />
    </logger>
    <logger name="org.springframework.jdbc.core.StatementCreatorUtils">
      <level value="trace" />
    </logger>
    <logger name="com.poscoict.glueframework.dao.manager.GlueQueryLoader">
        <level value="trace" />
    </logger>
    <logger name="org.springframework">
        <level value="info" />
```



절대경로 사용시 전체 경로를 지정한다.
directory가 없다면 생성한다.

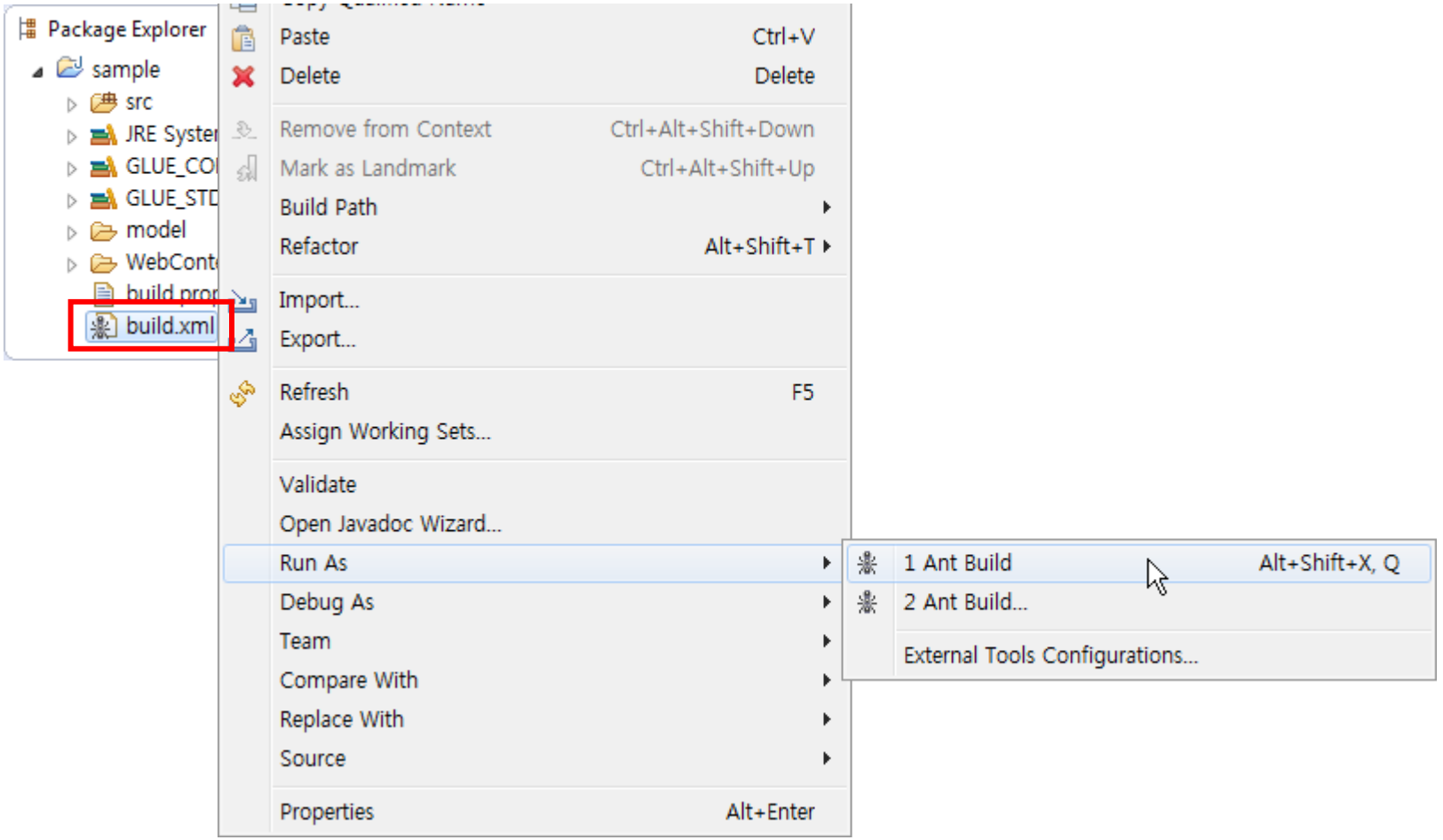
- 실습#1-4
  - applicationContext.xml 수정
    - DB연결 정보 설정



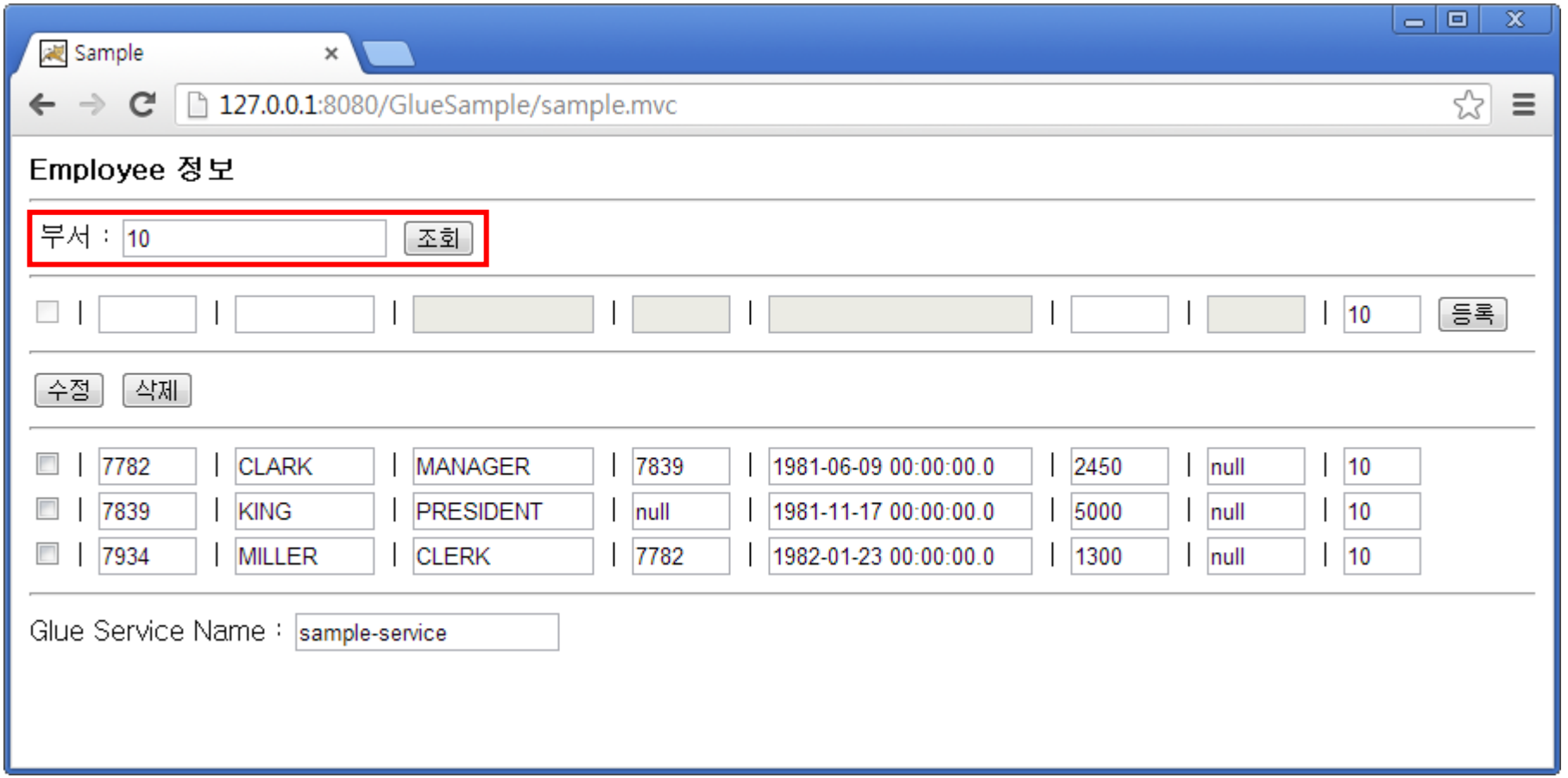
build.properties | logback-test.xml | *applicationContext.xml

```
<?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/spring-

    <bean id="serviceManager" class="com.poscoict.glueframework.biz.control
        <property name="cacheManager" ref="cacheManager" />
        <property name="serviceLoader" ref="serviceLoader" />
    </bean>
    <bean id="cacheManager" class="com.poscoict.glueframework.cache.ehcache
    <bean id="serviceLoader" class="com.poscoict.glueframework.biz.control.

    <bean id="test-dao" class="com.poscoict.glueframework.dao.jdbc.GlueJdbc
        <property name="dataSource" ref="test-dataSource"/>
        <property name="queryManager" ref="queryManager"/>
    </bean>
    <bean id="test-tx" class="com.poscoict.glueframework.transaction.GlueDa
        <property name="dataSource" ref="test-dataSource"/>
    </bean>
    <bean id="test-dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
        <property name="driverClassName" value="org.sqlite.JDBC"/>
        <property name="url" value="jdbc:sqlite:/C://apache-tomcat-7.0.57/sample.db"/>
        <property name="defaultAutoCommit" value="false"/>
        <property name="minIdle" value="0"/>
        <property name="maxActive" value="-1"/>
        <property name="maxIdle" value="1000"/>
    </bean>
    <bean id="queryManager" class="com.poscoict.glueframework.dao.manager.GlueQueryManagerImpl">
        <property name="cacheManager" ref="cacheManager"/>
        <property name="queryLoader" ref="queryLoader"/>
    </bean>
```

- 실습#1-5
  - Deploy
    - Build script 실행



- 실습#1-6
  - 화면실행
    - http://127.0.0.1:8080/GlueSample/sample.mvc



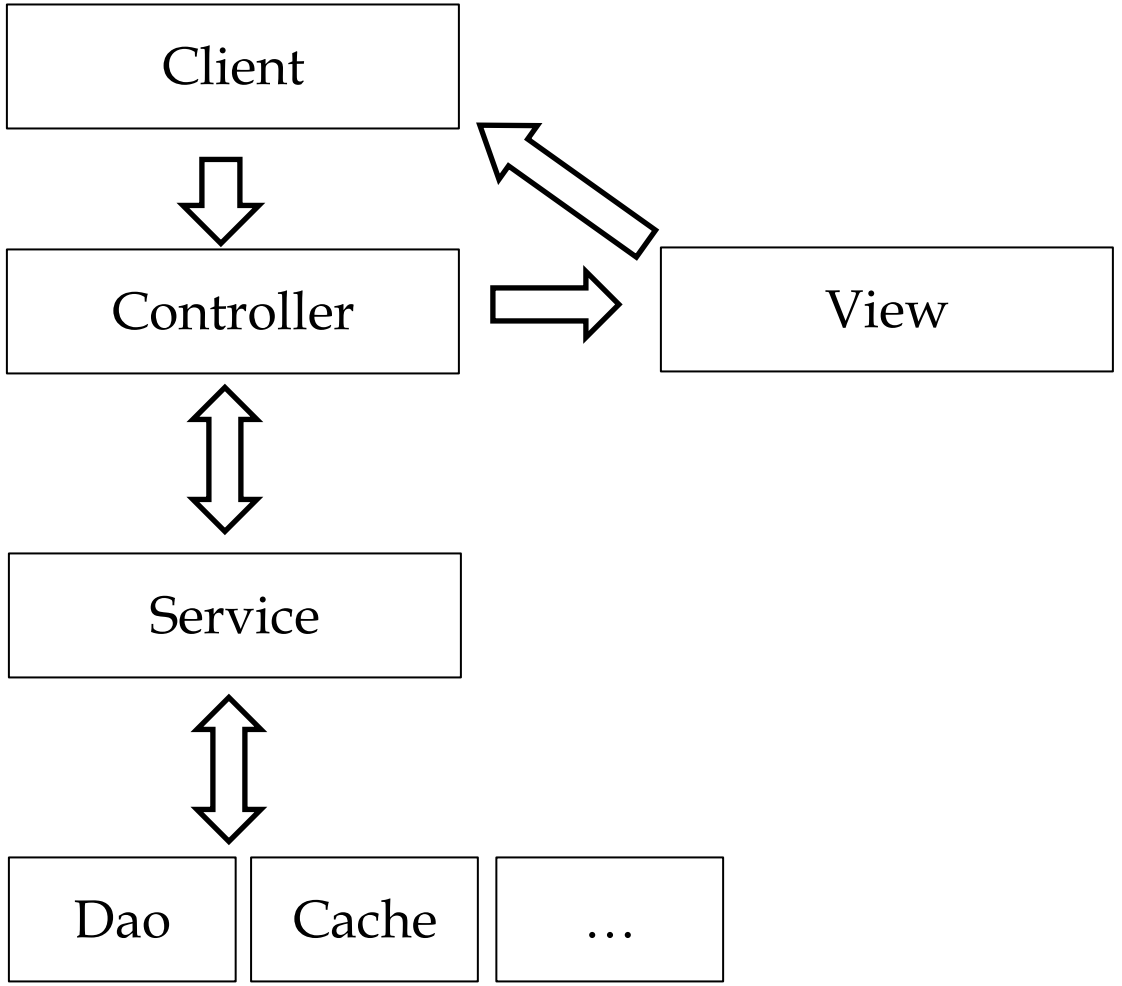
Glue Framework 설계 가이드

개발 프로세스 정의

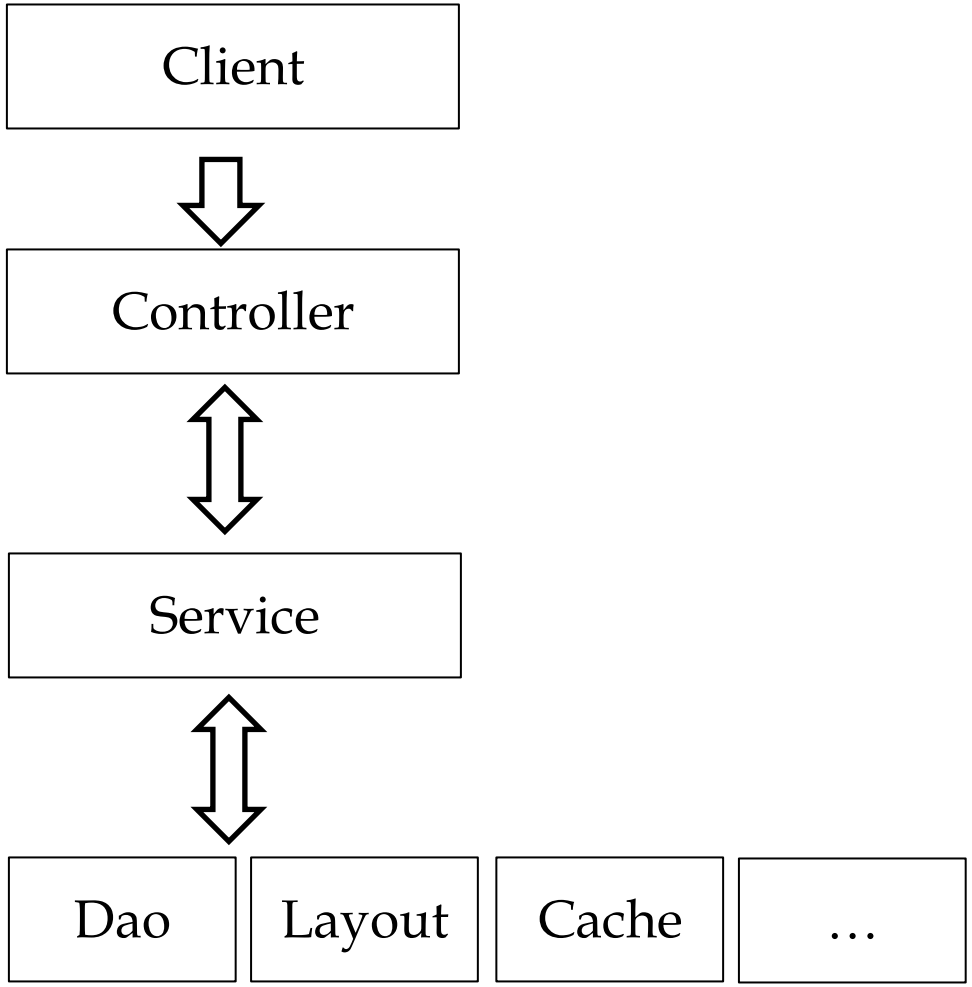
단계	주요 수행 업무	주요 산출물
분석	현행 업무 및 시스템을 분석하고, 사용자 요구사항을 도출하며, 이를 토대로 업무 프로세스, 이벤트 모델링, 보안 관련 분석을 함	고객요구정의서, 현행시스템 분석서, 시스템 아키텍처 등
설계	사용자 요구사항 및 분석 결과에 근거하여 업무기능, 사용자 인터페이스 및 내/외부 인터페이스 등을 구현 가능한 수준으로 설계	업무, 어플리케이션, 화면, 시스템 인터페이스, DB, 보완, 데이터 이행 설계서 Page Flow Diagram, Activity Diagram 화면 입출력사양서, 화면 Layout (HTML) Query정의서, 전문Layout
구현	설계에 따라 응용시스템 기능의 충분성, 완전성, 무결성, 편의성, 적정성을 확보할 수 있도록 구현하고, 단위 기능에 대한 검증을 수행	프로그램 Source Code, 단위테스트 계획서 및 결과서
시험	통합시험, 시스템 시험을 통하여 구현된 시스템이 통합관점에서의 기능 완전성과 성능, 안정성, 보완성 확보 여부를 검증	결합/시스템/운영 테스트 계획서 및 결과서, 운영교육 계획서 및 결과서, 사용자/시스템 운영 매뉴얼
전개	시스템을 운영하기 위한 시스템 설치 및 배포, 초기 데이터 구축 등의 준비를 완료하고, 시스템이 사용자에게 이관 및 운영될 수 있도록 준비	이행 계획서 및 결과서

※ 위 개발프로세스는 권고사항이며, 프로젝트의 개발 방법론 및 **PM**에 따라서 변경 가능함

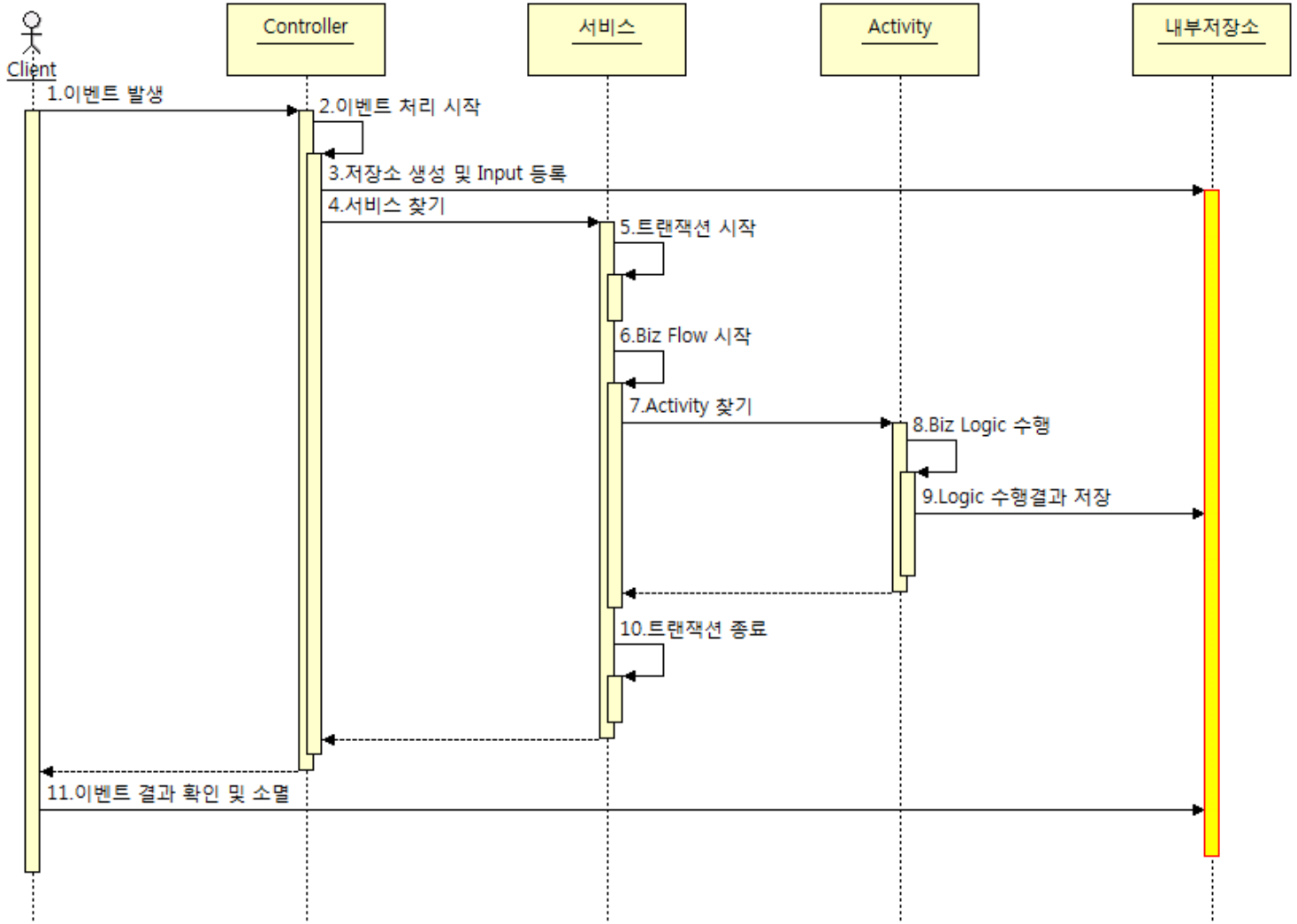
- **화면 Flow**
  - **Client : 브라우저**
  - **Controller : Servlet**
    - 2가지 Type 제공
      - struts 1
      - spring mvc
    - web.xml
  - **Service**
    - Glue Activity Diagram의 결과물
    - Biz Controller에 의해 Activity들의 정의된 flow대로 수행된다.
  - **DAO**
    - GlueGenericDao
    - Spring JDBC, Hibernate, iBatis
  - **View**
    - jsp 등에 기반에 client에서 결과를 보여즘.



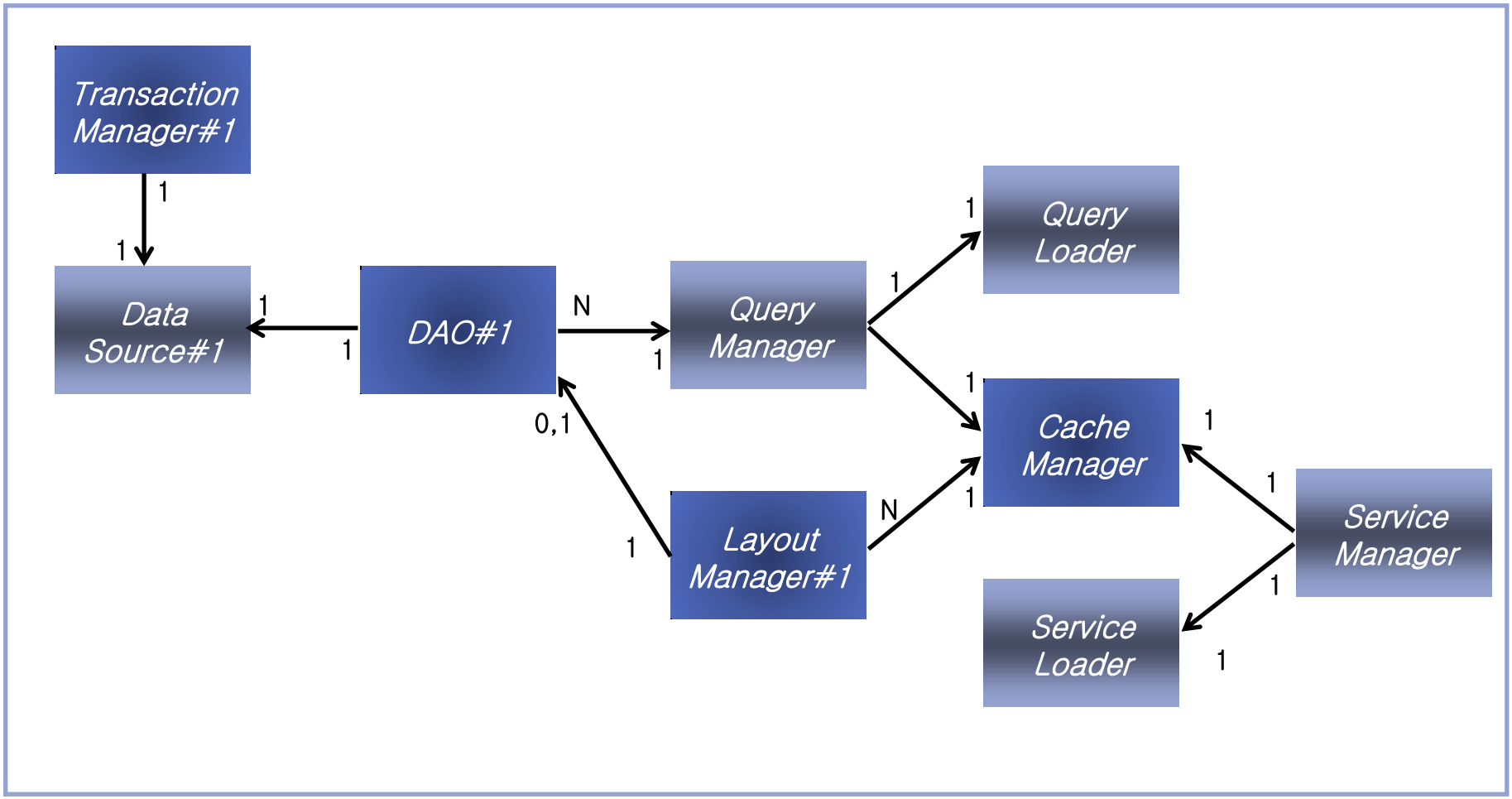
- **TC 처리 Flow**
  - **Client : EAI 등**
  - **Controller : Servlet 등**
    - HttpReceiver
    - web.xml
  - **Service**
    - Glue Activity Diagram의 결과물
    - Biz Controller에 의해 Activity들의 정의된 flow대로 수행된다.
  - **DAO**
    - GlueGenericDao
    - SpringJDBC, Hibernate, iBatis
  - **Layout**
    - GlueMessageLayout
    - Message 생성, 파싱 담당



내부저장소(GlueContext)



- DAO, Layout 관련 Bean
  - A→B: A가 B를 참조한다(Referencing)



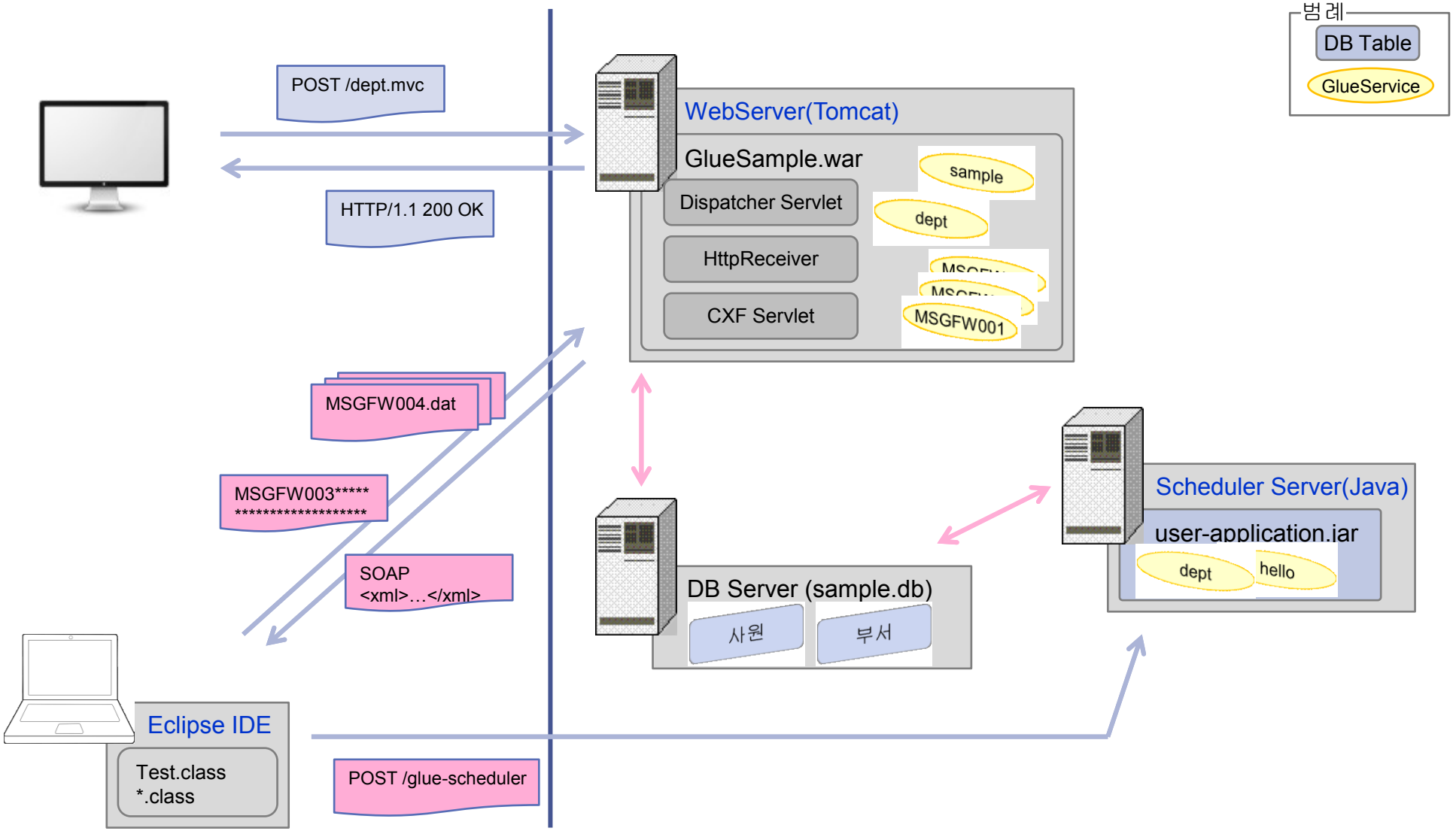
설계 산출물

UI(Web) Program

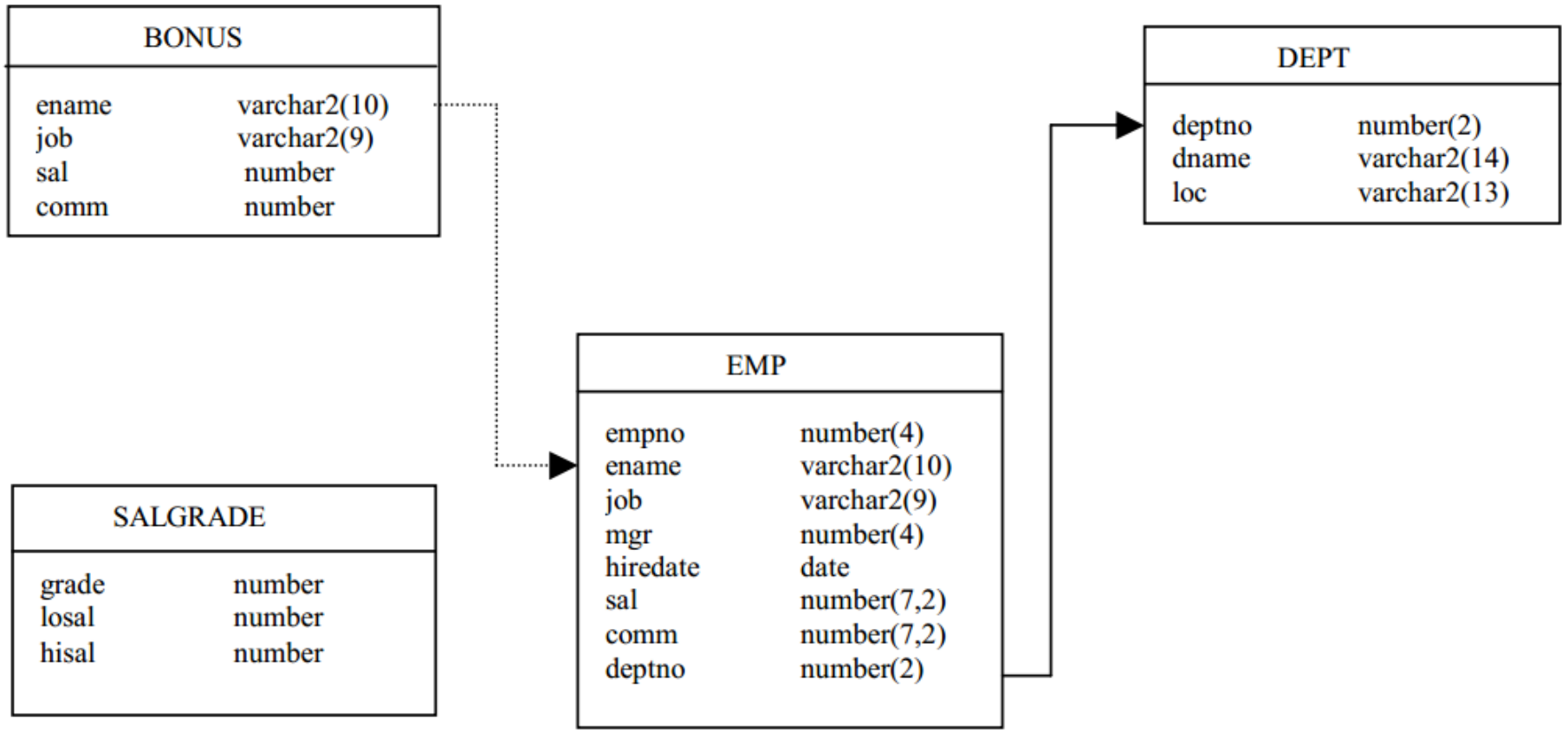
- 화면 (Html/JSP) 정의서
- I/O 정의서
- xx-query.glue_sql
- Activity Diagram

NonUI(TC) Program

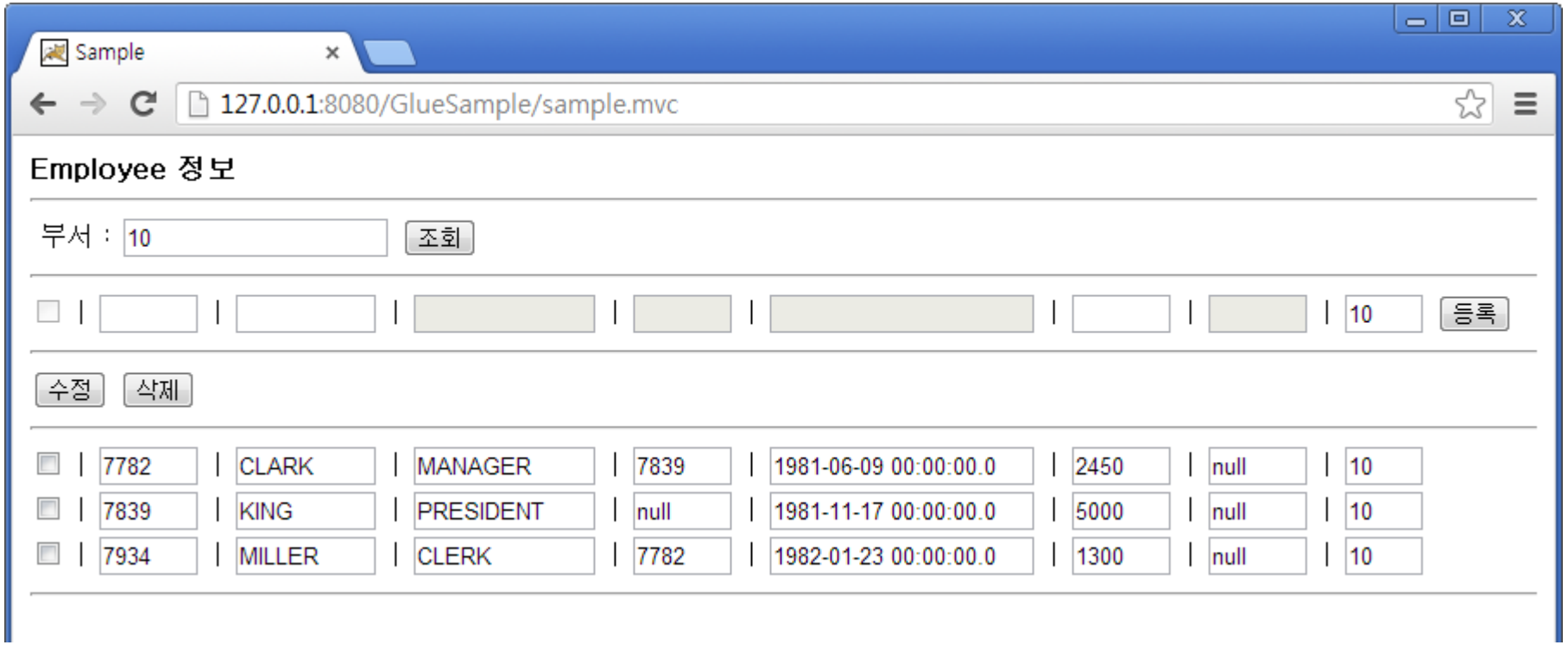
- Layout 정의서
- xx-query.glue_sql
- Activity Diagram


범례
DB Table
GlueService
POST /dept.mvc
HTTP/1.1 200 OK
WebServer(Tomcat)
GlueSample.war
Dispatcher Servlet
HttpReceiver
CXF Servlet
sample
dept
MSGFW001
MSGFW004.dat
MSGFW003*****
****
SOAP
<xml>…</xml>
DB Server (sample.db)
사원
부서
Scheduler Server(Java)
user-application.jar
dept
hello
Eclipse IDE
Test.class
*.class
POST /glue-scheduler

DB 설계 결과


BONUS
ename varchar2(10)
job varchar2(9)
sal number
comm number
DEPT
deptno number(2)
dname varchar2(14)
loc varchar2(13)
EMP
empno number(4)
ename varchar2(10)
job varchar2(9)
mgr number(4)
hiredate date
sal number(7,2)
comm number(7,2)
deptno number(2)
SALGRADE
grade number
losal number
hisal number

화면 설계 결과



- deptno : 검색조건
- chk_insert , empno_insert, ename_insert, sal_insert, deptno_insert : 등록
- chk, EMPNO, ENAME, JOB,MGR,HIREDATE,SAL,COMM,DEPTNO : 수정/삭제
- find, insert, update, delete : 버튼

인터페이스 설계 결과

요건정의

송신	수신	인터페이스	… … …
시스템#1	시스템#2	TC : 부서정보	…
시스템#1	시스템#3	FILE : 사원정보	…
시스템#2	시스템#4	TC : 부서관리자정보	…
…	…	…	…

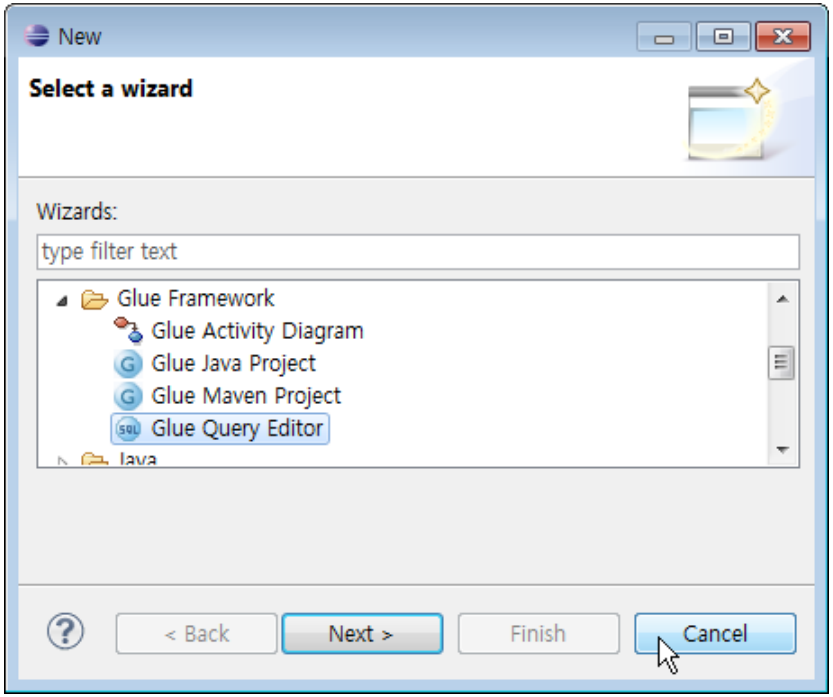
Layout 정의

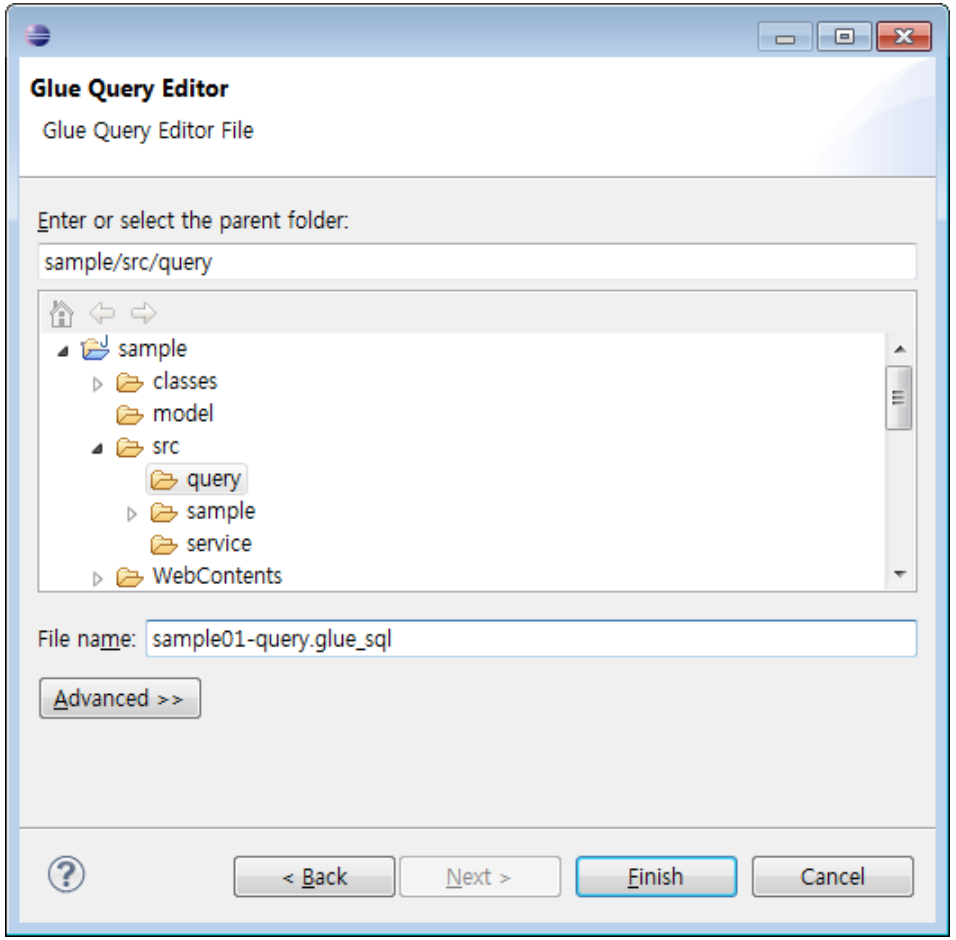
부서정보 : MSGFW001				
항목명	항목ID	Data Type	길이	비고
CRUD구분	CRUD	Stirng	1	처리타입 정의, - C:추가, S:송신, U:수정,D:삭제
부서번호	DEPTNO	Number	2	…
부서이름	DNAME	String	14	…
부서위치	LOC	String	13	…

- Query 관리 파일
  - 자바 소스 코드 내 하드 코딩된 SQL 을 지양함
  - 쿼리의 가독성과 관리 편리성 향상
  - 작명 규칙
    - xxx-query.glue_sql
  - 애플리케이션 상의 위치
    - query \
      - jar:file:GlueSample.war! \WEB-INF \ classes \ query
      - jar:file:user-application.jar! \ query
  - Glue Project 상의 위치
    - src \ query
      - <sample> \ src \ query
      - <sample-maven> \ src \ main \ resource \ query

Glue Query Editor 생성

- File → New → other → Glue Framework → Glue Query Editor 선택





Glue Query Editor 구성 내용

queryMap

- desc : 쿼리파일 성격에 맞는 알맞은 설명을 기재

query

- id : 쿼리를 대표할 수 있는 유일한 명칭 기재.
  전체 쿼리 파일에서 유일한 값이어야 함.
- desc : 쿼리의 부가적인 설명을 기재
- returnType : 한 개의 row를 표현하는 Value Object
- isNamed : SQL안에 binding 값에 변수 명을 기재하는 유형의 SQL인지.
  ? 를 사용하는지 :name 을 사용하는지

```
<?xml version="1.0" encoding="UTF-8"?>
<queryMap desc="sample query" ...>
    <query id="emp.select" desc="EMP 조회" resultType="" isNamed="false">
        <![CDATA[
select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
from EMP
where DEPTNO=?
        ]]>
    </query>
...
```

- **실습#2**
  - **sample01-query.glue_sql : ? 를 사용하는 쿼리**
    - sample01.emp.select
    - sample01.emp.update
    - sample01.emp.delete
    - sample01.emp.insert
  - **sample02-query.glue_sql : named 쿼리**
    - sample02.emp.select
    - sample02.emp.update
    - sample02.emp.delete
    - sample02.emp.insert
  - **dept-query.glue_sql**
    - dept.select
    - dept.insert
    - dept.update
    - dept.delete

실습#2-1

sample01-query.glue_sql : ? 를 사용하는 쿼리

```
<queryMap desc="sample01" ...>
    <query id="sample01.emp.select" desc="EMP 조회" resultType="">
        <![CDATA[

select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO

from EMP

where DEPTNO=?

        ]]>
    </query>
    <query id="sample01.emp.update" desc="EMP 수정" resultType="" >
        <![CDATA[

update emp

set sal=?, ename=?

where empno=?

        ]]>
    </query>
```

실습#2-1

sample01-query.glue_sql : ? 를 사용하는 쿼리

```
    <query id="sample01.emp.delete" desc="EMP 삭제" resultType="">
        <![CDATA[
delete from emp
where empno=?
        ]]>
    </query>
    <query id="sample01.emp.insert" desc="EMP 등록" resultType="">
        <![CDATA[
insert into emp(empno, ename, sal, deptno)
values(?, ?, ?, ?)
        ]]>
    </query>
</queryMap>
```

실습#2-2

sample02-query.glue_sql : named 쿼리

```
<queryMap desc="sample02" ...>
  <query id="sample02.emp.select" desc="" resultType="" isNamed="true" >
        <![CDATA[

select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO

from EMP

where DEPTNO=:deptno

        ]]>
  </query>
  <query id="sample02.emp.update" desc="" resultType="" isNamed="true">
        <![CDATA[

update emp

set sal=:sal, ename=:ename

where empno=:empno

        ]]>
  </query>
```

실습#2-2

sample02-query.glue_sql : named 쿼리

```
  <query id="sample02.emp.delete" desc="" resultType="" isNamed="true">
      <![CDATA[
delete from emp
where empno=:empno
      ]]>
  </query>
  <query id="sample02.emp.insert" desc="" resultType="" isNamed="true">
      <![CDATA[
insert into emp(empno, ename, sal, deptno)
values(:empno, :ename, :sal, :deptno)
      ]]>
  </query>
</queryMap>
```

실습#2-3

dept-query.glue_sql

```
<queryMap desc="sample03" ...>
  <query id="dept.select" desc="" resultType="" isNamed="false" >
        <![CDATA[

select *

from DEPT

        ]]>
  </query>
  <query id="dept.update" desc="" resultType="" isNamed="false">
        <![CDATA[

update DEPT

set DNAME=?, LOC=?

where DEPTNO=?

        ]]>
  </query>
```

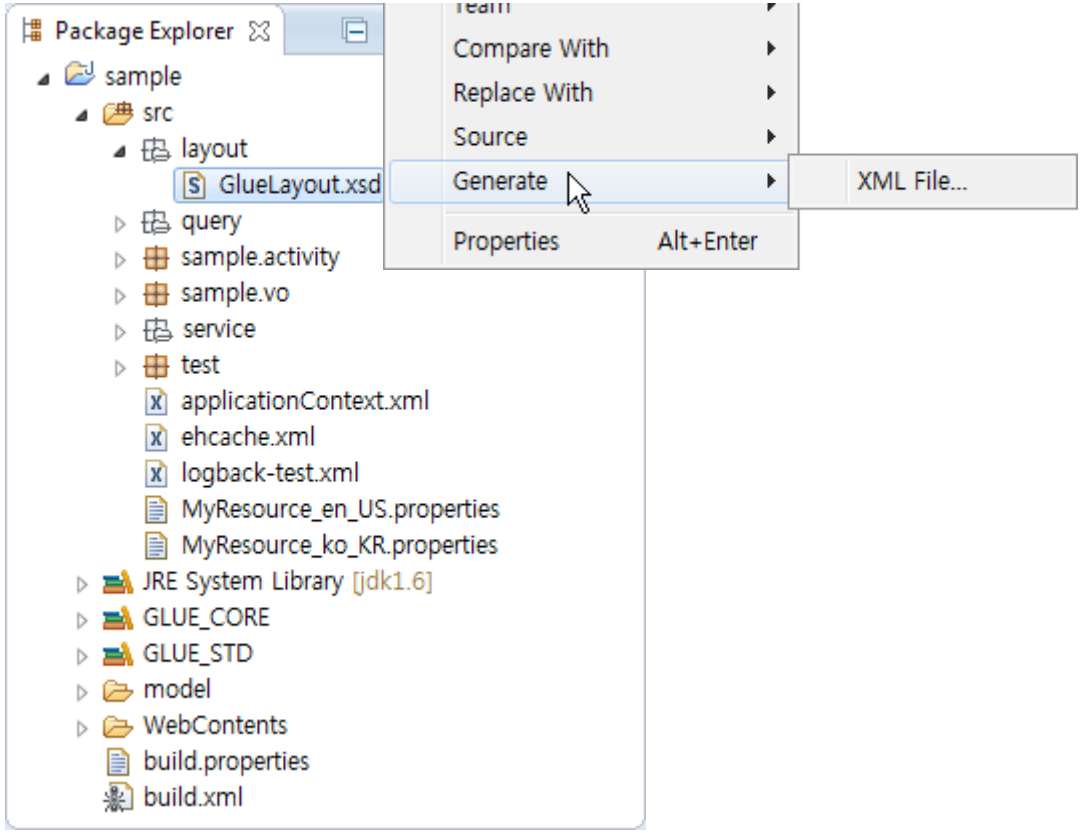
실습#2-3

dept-query.glue_sql

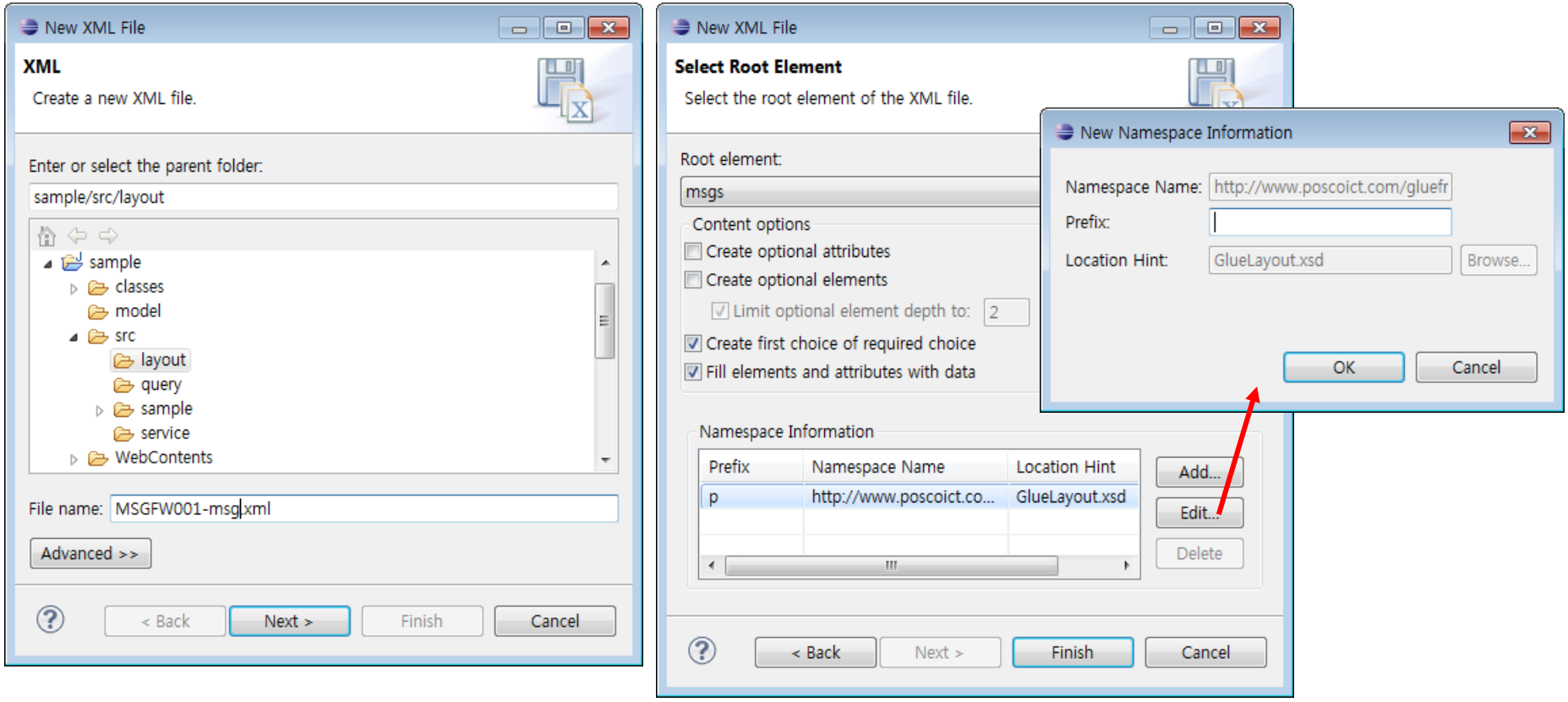
```
  <query id="dept.delete" desc="" resultType="" isNamed="true">
      <![CDATA[
delete from DEPT
where DEPTNO=:deptno
      ]]>
  </query>
  <query id="dept.insert" desc="" resultType="" isNamed="false">
      <![CDATA[
insert into DEPT(DEPTNO, DNAME, LOC)
values(?, ?, ?)
      ]]>
  </query>
</queryMap>
```

- **Message Layout 파일**
  - **TC(전문)의 Data 구성을 정의한 파일, Interface Layout**
  - **TC를 파싱, 생성할 때 기준 정보로 사용 됨**
  - **TC별로 작성**
  - **작명 규칙**
    - TCID-msg.xml
  - **애플리케이션 상의 위치**
    - layout\
      - GlueSample.war! \WEB-INF \ classes \ layout
      - user-application.jar! \ layout
  - **Glue Project 상의 위치**
    - src\layout
      - <sample> \ src \ layout
      - <sample-maven> \ src \ main \ resource \ layout

- **Message Layout XML 파일 생성**
  - **XML Schema를 통해 XML를 생성함.**



- Message Layout XML 파일 생성
  - XML Schema를 통해 XML를 생성함.
  - Prefix 는 제거함



Message Layout 구성 내용

- **msg**
  - id : Message Layout에 대한 TC ID를 의미
    ID는 파일명에 사용됨
  - name : Message 명 및 Message 설명을 정의
- **attribute**
  - type : 항목이 일반 항목인지 그룹항목인지 여부를 정의
    - E, G , GE
  - seq : 항목의 순서를 정의
  - id : 항목ID
  - name : 항목명
  - datatype : 항목의 데이터 타입
    - STRING, NUMBER, DATE, ARRAY
  - length : 항목의 총 길이.
    Type이 G인 경우는 반복 회수를 의미
  - precision : datatype이 NUMBER type일 경우 소수점 자리수

- 실습#3
  - **MSGFW001-msg.xml : 일반 항목 사용**
    - TRANSACTION_CODE
    - CRUD
    - DEPTNO
    - DNAME
    - LOC
  - **MSGFW002-msg.xml : 그룹 항목 사용**
    - TRANSACTION_CODE
    - CRUD
    - DEPTNO
    - DNAME
    - LOC
    - EMPNO : 그룹 항목(3회 반복)
    - ENAME : 그룹 항목(3회 반복)
    - ETC

실습#3-1

MSGFW001-msg.xml : 일반 항목 사용

```
<?xml version="1.0" encoding="EUC-KR"?>
<msgs ...>
  <msg id="MSGFW001" name="DEPT">
    <attribute type="E" seq="1"
          id="TRANSACTION_CODE" name="TransactionCode"
          datatype="STRING" length="8" />
    <attribute type="E" seq="2"
          id="CRUD" name="CRUD"
          datatype="STRING" length="1" />
    <attribute type="E" seq="3"
          id="DEPTNO" name="부서번호"
          datatype="NUMBER" length="2" precision="0" />
    <attribute type="E" seq="4"
          id="DNAME" name="부서명"
          datatype="STRING" length="14" />
    <attribute type="E" seq="5"
          id="LOC" name="부서위치"
          datatype="STRING" length="13" />
  </msg>
</msgs>
```

실습#3-2

MSGFW002-msg.xml : 그룹 항목 사용

```
<?xml version="1.0" encoding="EUC-KR"?>
<msgs ...>
  <msg id="MSGFW002" name="DEPT">
    <attribute type="E" seq="1"
          id="TRANSACTION_CODE" name="TransactionCode"
          datatype="STRING" length="8" />
    <attribute type="E" seq="2"
          id="CRUD" name="CRUD"
          datatype="STRING" length="1" />
    <attribute type="E" seq="3"
          id="DEPTNO" name="부서번호"
          datatype="NUMBER" length="2" precision="0" />
    <attribute type="E" seq="4"
          id="DNAME" name="부서명"
          datatype="STRING" length="14" />
    <attribute type="E" seq="5"
          id="LOC" name="부서위치"
          datatype="STRING" length="13" />
```

실습#3-2

MSGFW002-msg.xml : 그룹 항목 사용

```
    <attribute type="G" seq="6"
          id="EMP_SET" name="부서소속 사원"
          datatype="ARRAY" length="3" />
    <attribute type="GE" seq="7"
          id="EMPNO" name="사원번호"
          datatype="NUMBER" length="4" precision="0" />
    <attribute type="GE" seq="8"
          id="ENAME" name="이름"
          datatype="STRING" length="10" />
    <attribute type="E" seq="9"
          id="ETC" name="기타"
          datatype="STRING" length="1" />
  </msg>
</msgs>
```

- Activity 란?
  - Activity는 단위 기능을 수행하며 GlueActivity를 상속 받은 Class 이다.
  - Activity는 Service 뿐만이 아니라 다른 Service에서도 공유 할 수 있으므로 재사용 가능하게 설계 하는 것이 좋다.

- Activity 예제
  - Activity는 Name과 Class로 이루어져 있고 다음 Activity로 분기상태인 Transition을 가진다. 또한 property를 정의하여 Class에서 사용 할 수 있다.

```
<activity name="InitProcess" class="sample.edu.FindEmpAC">
    <transition name="success" value="EventRouter"/>
    <transition name="failure" value="HandleError"/>
    <property name="condition" value="current"/>
    <property name="deptno" value="10"/>
</activity>
```

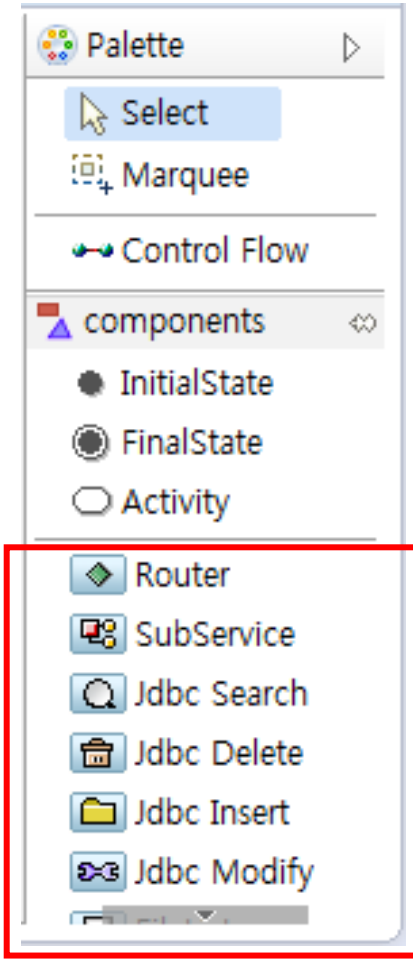
Activity의 구성 요소

- **name**
  - Activity의 Name은 Service 에서 유일하여야 한다.
- **class**
  - class는 반드시 GlueActivity Class를 상속 받은 Class이어야 한다.
- **transition**
  - Activity는 반드시 하나 이상의 transition을 가지고 있어야 하며 Router가 아닌 Activity는 반드시 "success" transition을 가지고 있어야 한다.
  - <transition name="success" value="EventRouter"/>
  - 만일 해당 Activity가 다음에 수행할 Activity가 존재하지 않는다면 <transition name="success" value="end"/> 로 지정하여야 한다.
  - success transition은 Activity Class가 Error 없이 정상 수행했을 경우이고 Error가 발생하면 failure transition이 수행 된다.
  - 만일 failure 가 정의 되지 않았다면 Service는 바로 종료 되고 Service를 호출한 Caller에게 Exception을 발생 하게 된다.
- **property**
  - property는 해당 Activity에서 사용할 수 있는 사용자 정의 Static Data이다.
  - 해당 Property에 개수 제한은 없고 필요한 만큼 선언 하여 사용할 수 있다.
  - 하나의 Activity내에서 같은 이름의 Property는 존재 할 수 없다.

5. 설계 – Activity Diagram

UML Activity 설명

Glue Framework에서 AD 작성시에는 다음 컴포넌트를 사용한다.



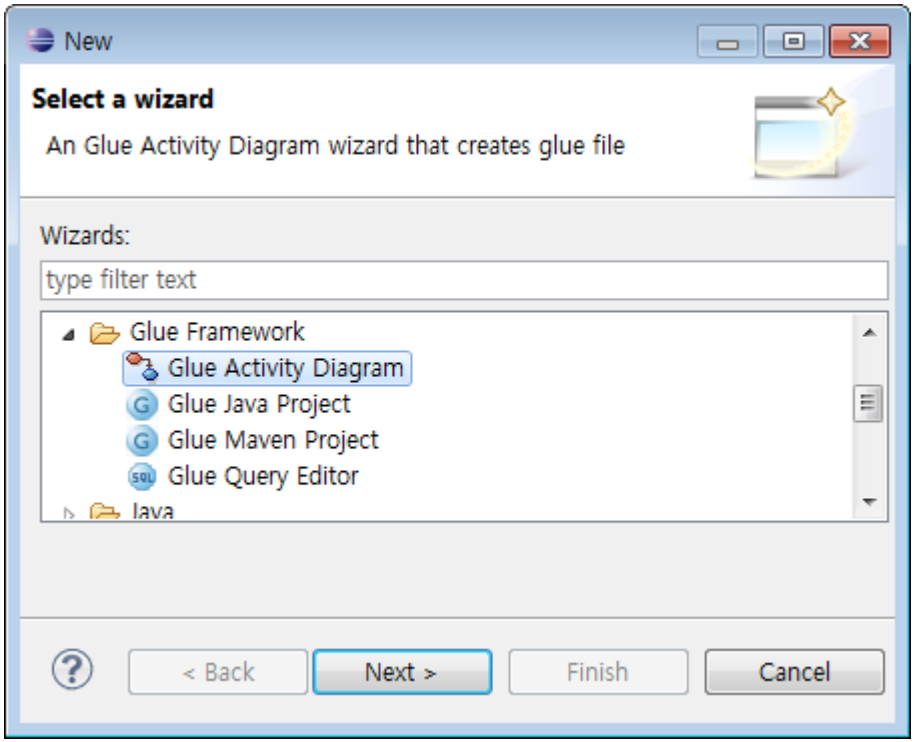
AD는 언제나 Initial State(●)로 시작하고 Final State(◉)로 종료된다.

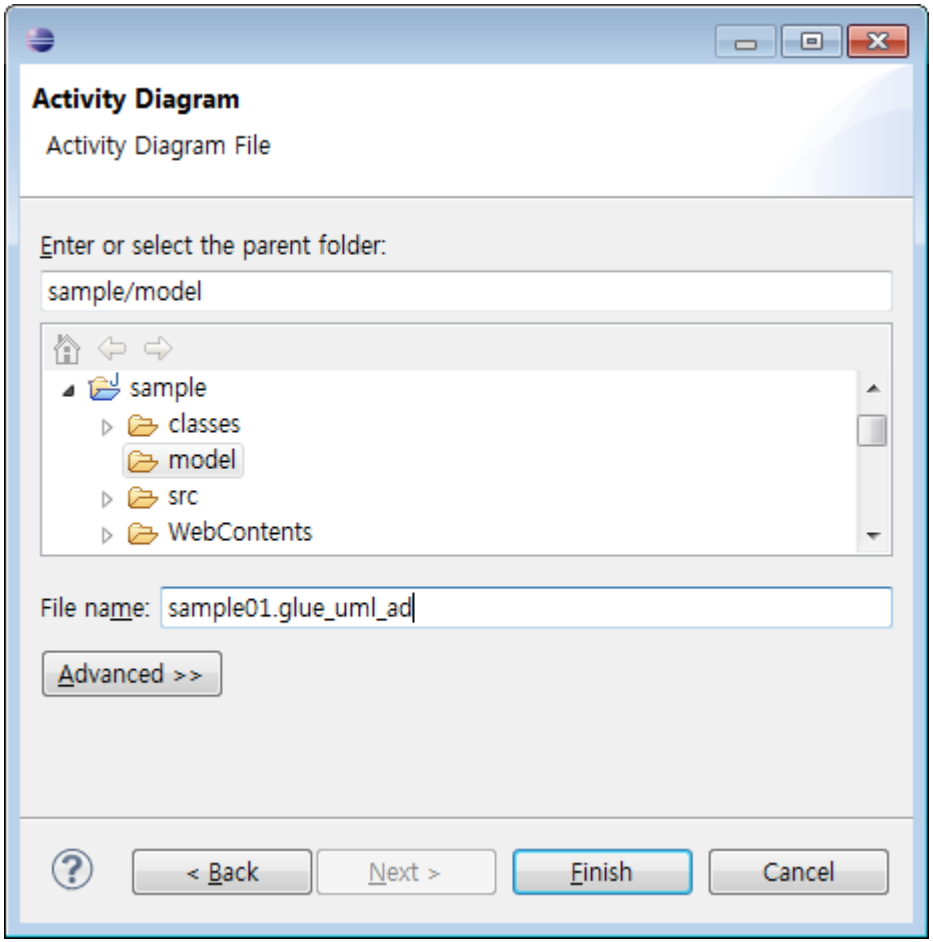
Control Flow : Transition, Activity간의 흐름을 표현한다.

Initial State : AD의 시작 시점을 의미한다.
Final State : AD의 종료점을 의미한다.
Activity: Custom Activity (하얀색)

Reusable Activity: 개발이 필요 없고 각 기능을 정의하여 Logic을 수행한다. (14가지 색상 지원)

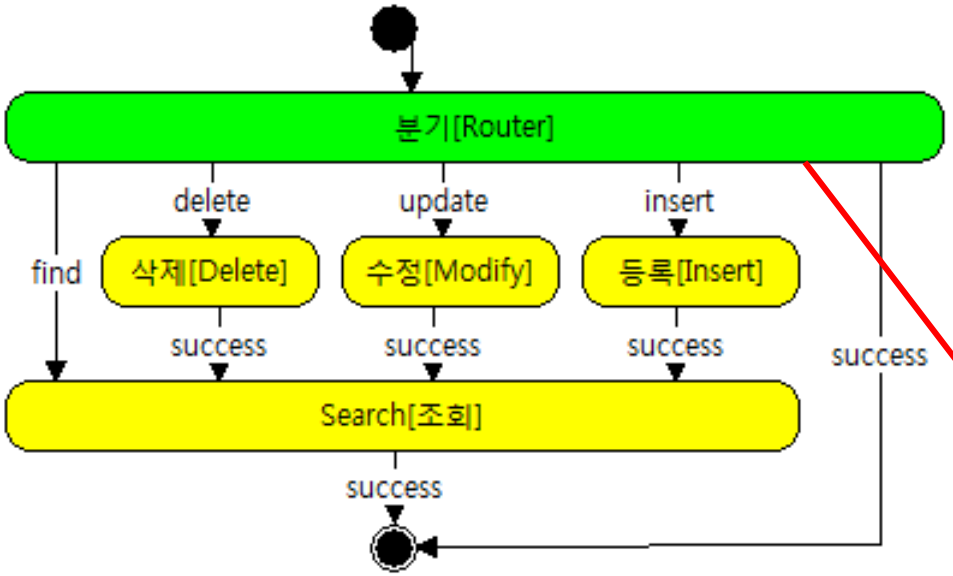
- Activity Diagram 생성
  - File → New → other → Glue Framework → Glue Activity Diagram 선택

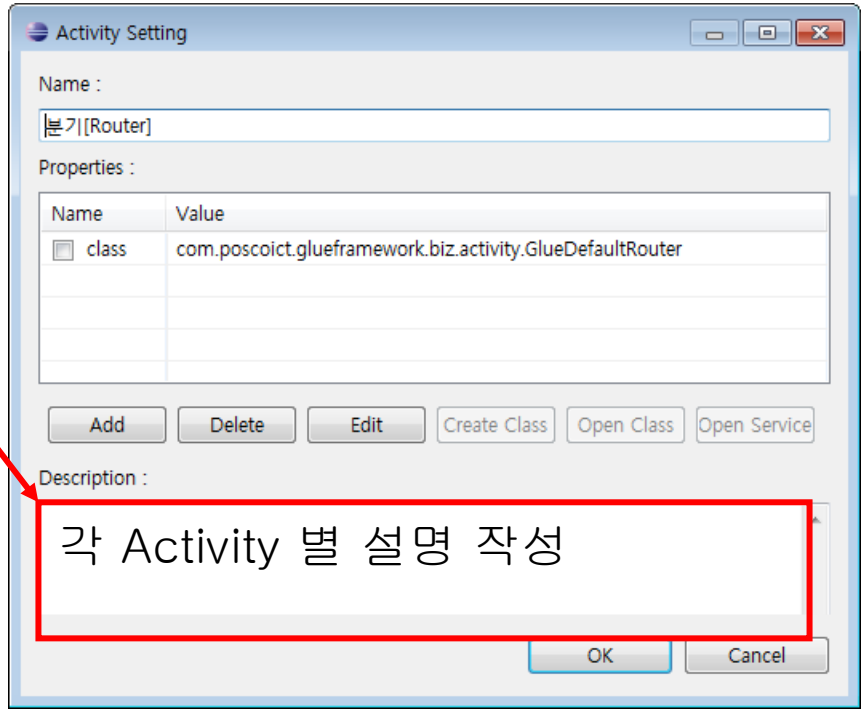




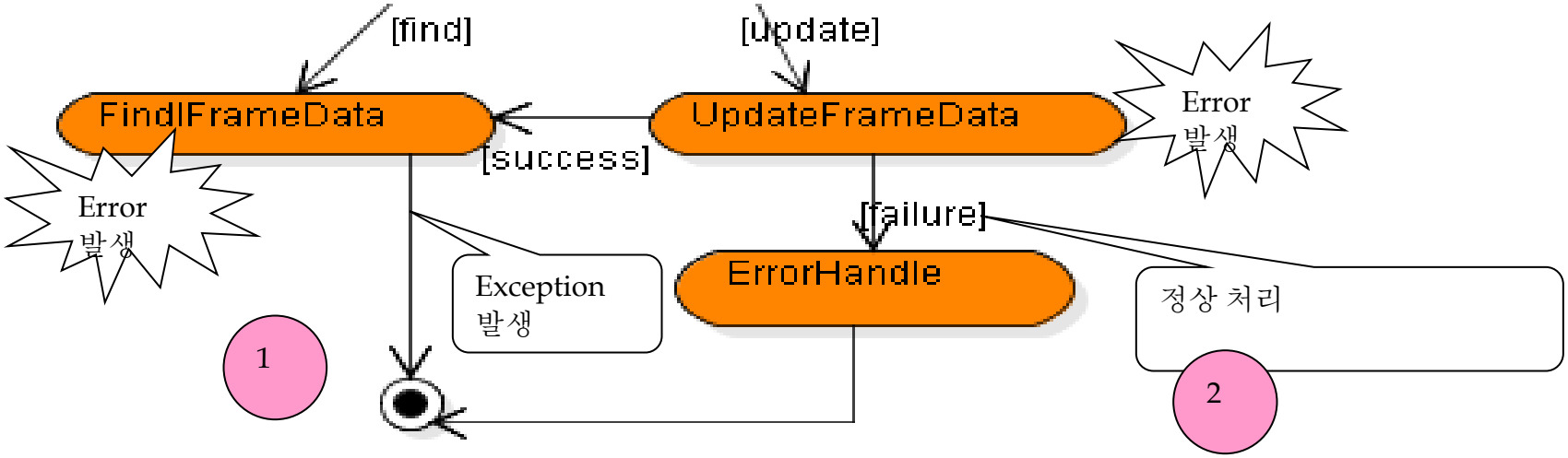
  - 서비스 명을 파일이름으로 기재한다.
    (확장자 : glue_uml_ad)
    - GLUEFW01(Non-UI)
    - emp(UI)

- Activity Diagram 설계
  - 요건 분석 결과를 반영 하여 Business Flow를 설계 한다.
  - Activity는 Business의 기능 단위로 도출 되어야 한다.
  - 기능 분석 결과와 I/O 정의서의 Event를 중심으로 아래와 같은 Activity Diagram을 작성할 수 있다.





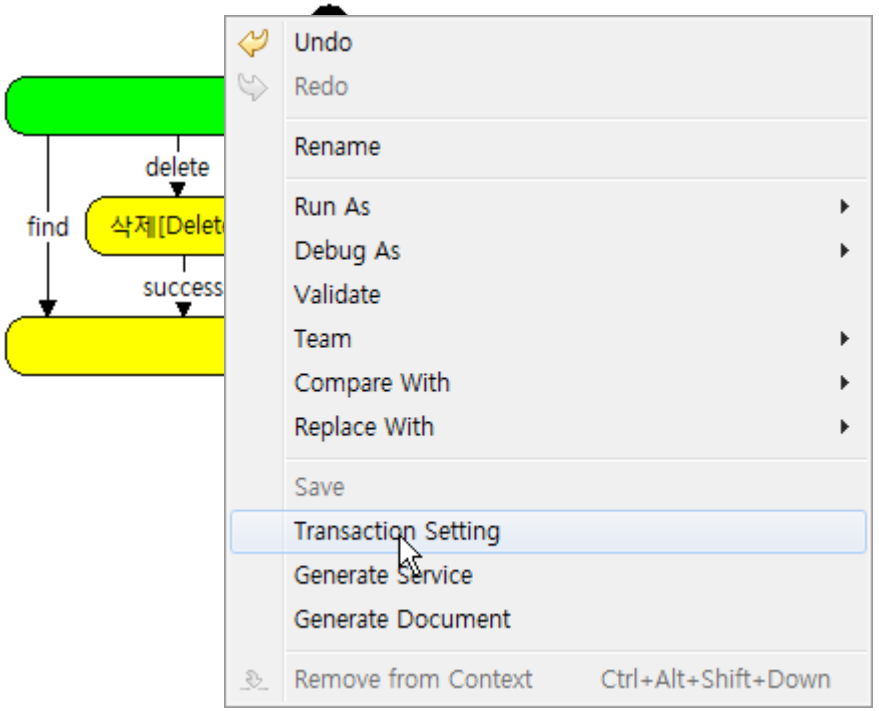
- **Error 처리**
  - **Case #1 : Error 처리를 하지 않은 경우**
    - 에러가 난 Activity에서 서비스 종료.
    - 상위 Layer (BizController) 에서 DB Transaction이 처리됨.
  - **Case #2 : Error 처리를 위한 Flow가 존재할 경우**
    - Error 처리를 위한 별도의 비즈니스가 있는 경우에 해당

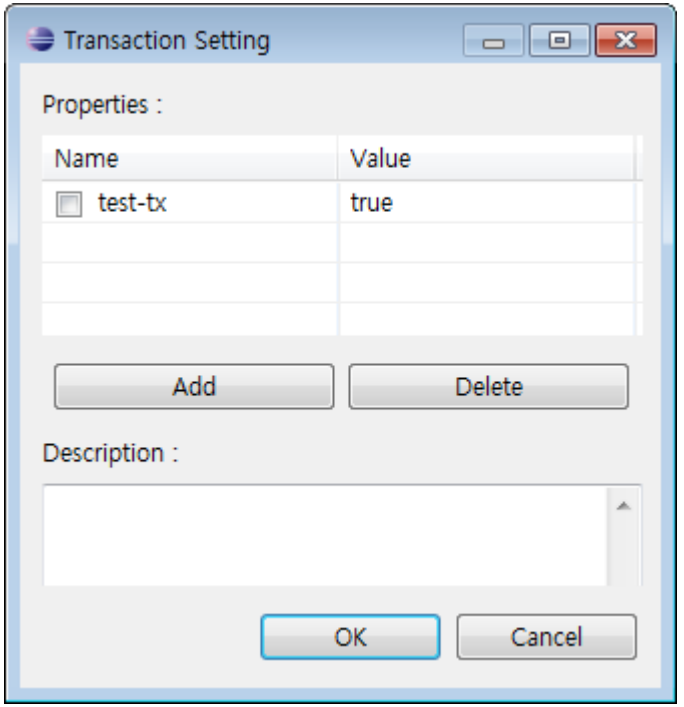


transaction-manager

```
<transaction-manager id="tx1" commit="true"/>
```

- **id**
  - Spring의 applicationContext.xml에 정의 되어 있는 Transaction Manager의 bean ID
- **Commit**
  - 해당 Transaction을 commit 할 지에 대한 여부

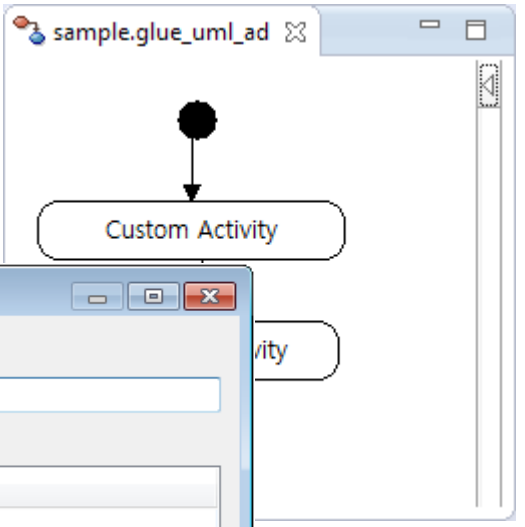


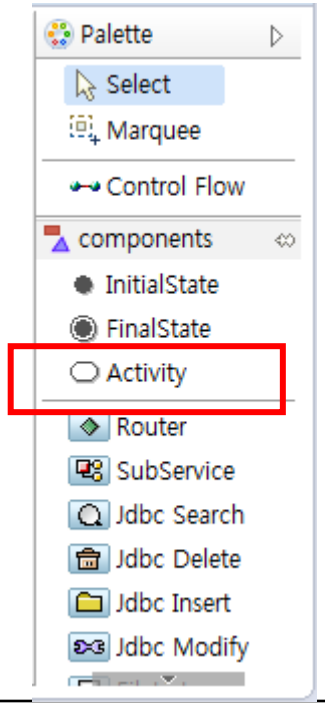


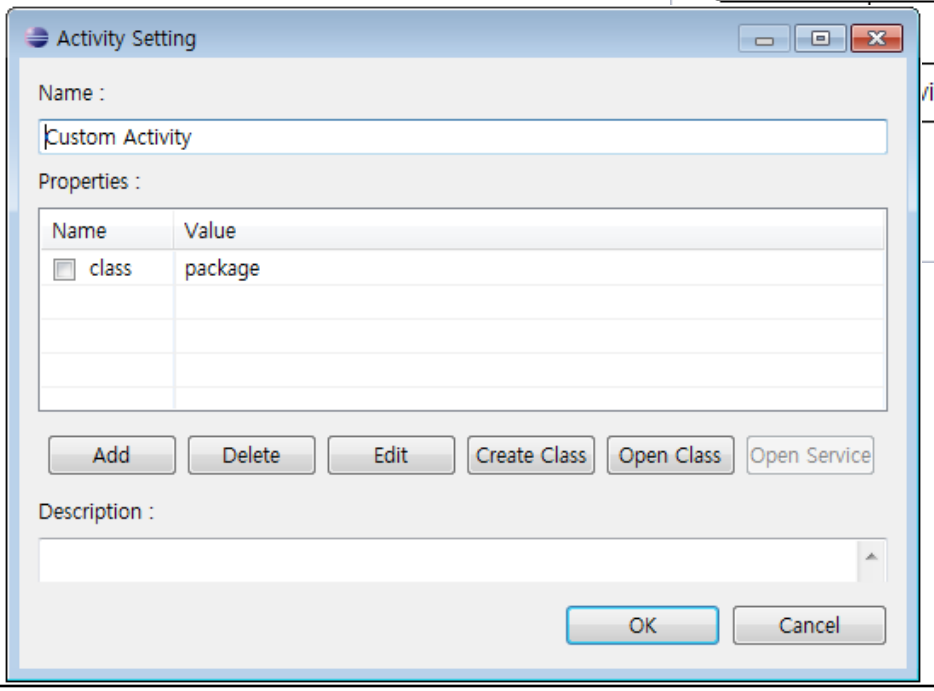
Custom Activity

```
<activity name="BizLogic" class="sample.edu.BizLogic">
    <transition name="success" value="end" />
</activity>
```

- 흰색
- property 편집 버튼 (Add, Delete) 사용가능
- 소스 편집 버튼 (Create Class, Open Class) 사용가능







- **실습#4**
  - **sample01.glue_uml_ad**
    - DAO를 통한 CRUD → test-dao, test-tx
    - ? 를 사용하는 SQL 사용 (sample01-query.glue_sql)
  - **sample02.glue_uml_ad**
    - DAO를 통한 CRUD → test-dao, test-tx
    - Named SQL 사용 (sample02-query.glue_sql)
  - **MSGFW001.glue_uml_ad**
    - XML Layout Manger 를 통한 Parsing → layoutInXml
  - **I/O 정의서 : sample01, sample02**
    - Event : find , insert , update , delete
    - 조회 : deptno
    - 등록 : empno_insert, ename_insert, sal_insert , deptno_insert,
    - 수정/등록 : chk, EMPNO, ENAME, SAL

실습#4-1

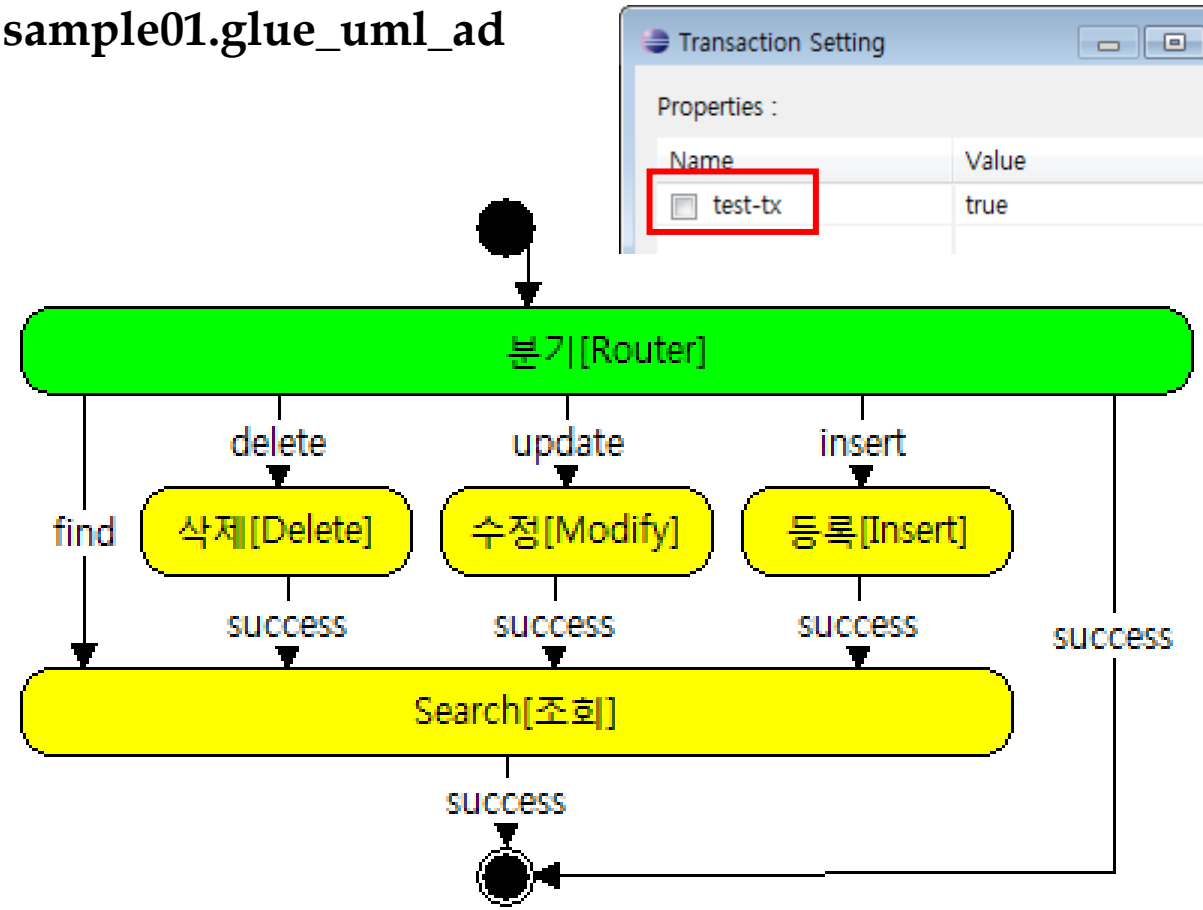
applicationContext.xml

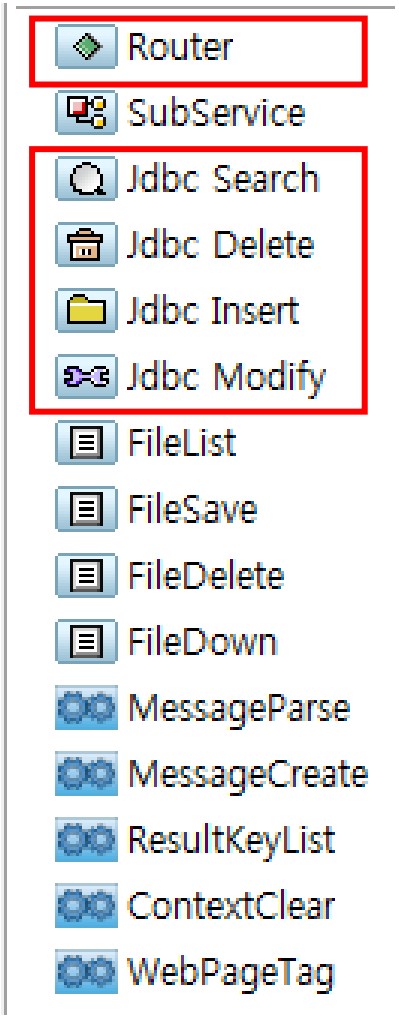
applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
    <bean id="test-dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
    <bean id="test-tx" class="com.poscoict.glueframework.transaction.GlueDataSourceTransactionManager">
    <bean id="test-dao" class="com.poscoict.glueframework.dao.jdbc.GlueJdbcDao">
    # JNDI
    <!--Mybatis Setting Start-->
    <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    <!--Mybatis Setting End-->
    <!--hibernate Setting Start-->
    <bean id="hibernateDao" class="com.poscoict.glueframework.dao.hibernate.GlueHibernateDao">
    <!--hibernate Setting End-->
    <bean id="cacheManager" class="com.poscoict.glueframework.cache.ehcache.GlueEhCacheManager"/>
    <bean id="cacheManager" class="com.poscoict.glueframework.cache.jcs.GlueJCSCacheManager"/>
    <bean id="serviceManager" class="com.poscoict.glueframework.biz.control.GlueServiceManagerImpl" lazy-init="true">
    <bean id="serviceLoader" class="com.poscoict.glueframework.biz.control.GlueServiceLoader">
    <bean id="queryManager" class="com.poscoict.glueframework.dao.manager.GlueQueryManagerImpl">
    <bean id="queryLoader" class="com.poscoict.glueframework.dao.manager.GlueQueryLoader">

    <bean id="messageSource" class="org.springframework.context.support.ResourceBundleMessageSource">
    <bean id="layoutInXml" class="com.poscoict.glueframework.message.layout.GlueXmlMessageLayout">
        <property name="cacheManager" ref="cacheManager"/>
        <property name="msgParsingType" value="byte"/>
    </bean>
    <bean id="layoutInDB" class="com.poscoict.glueframework.message.layout.GlueDBMessageLayout">
</beans>
```

Design Source

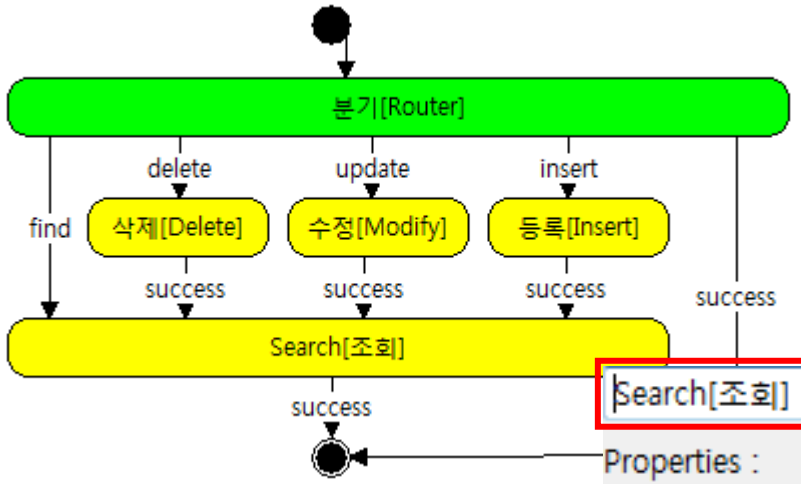
- 실습#4-2
  - sample01.glue_uml_ad





실습#4-2

sample01.glue_uml_ad



SQL

```
select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
from EMP
where DEPTNO=?
```

Search[조회]

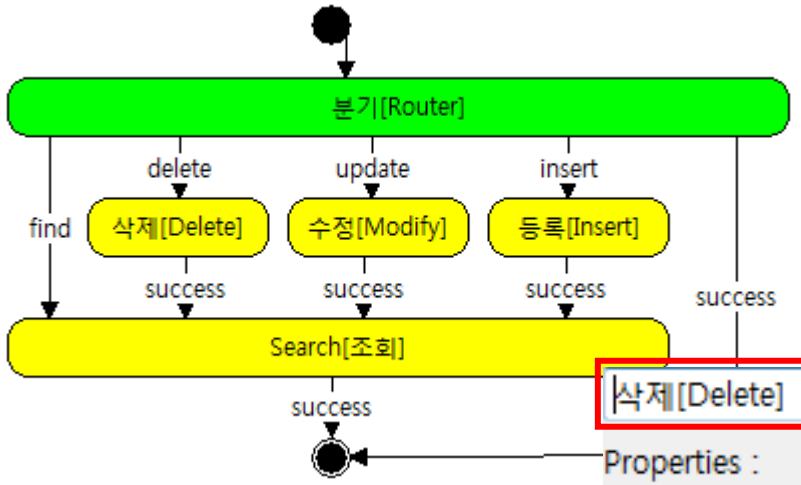
Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcSearch
dao	test-dao
param-count	1
result-key	EmpList
sql-key	sample01.emp.select
param0	deptno

- param-bindings 삭제
- cache-key 삭제
- cache-region 삭제
- cache-manager 삭제

실습#4-2

sample01.glue_uml_ad



SQL

```
delete from emp
where empno=?
```

삭제[Delete]

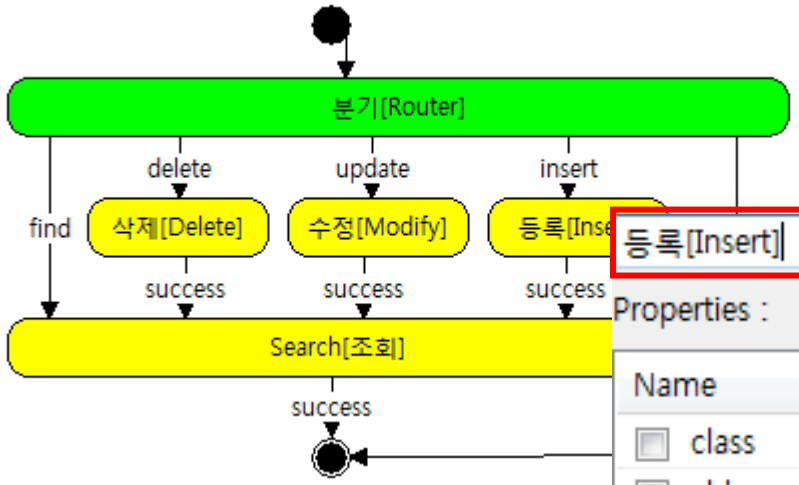
Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcDelete
chk-name	chk
dao	test-dao
param-count	1
sql-key	sample01.emp.delete
param0	EMPNO

- param-bindings 삭제
- list-key 삭제
- result-key 삭제

실습#4-2

sample01.glue_uml_ad



SQL

```
insert into emp(empno, ename, sal, deptno)
values(?, ?, ?, ?)
```

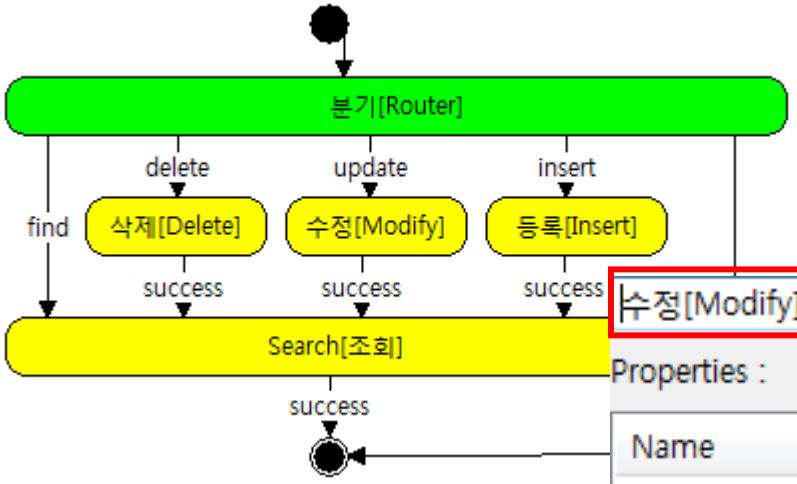
등록[Insert]

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcInsert
chk-name	chk_insert
dao	test-dao
param-count	4
result-key	insertCnt
sql-key	sample01.emp.insert
param0	empno_insert
param1	ename_insert
param2	sal_insert
param3	deptno_insert

- param-bindings 삭제
- list-key 삭제
- is-audit 삭제

- 실습#4-2
  - sample01.glue_uml_ad



SQL

```
update emp
set sal=?, ename=?
where empno=?
```

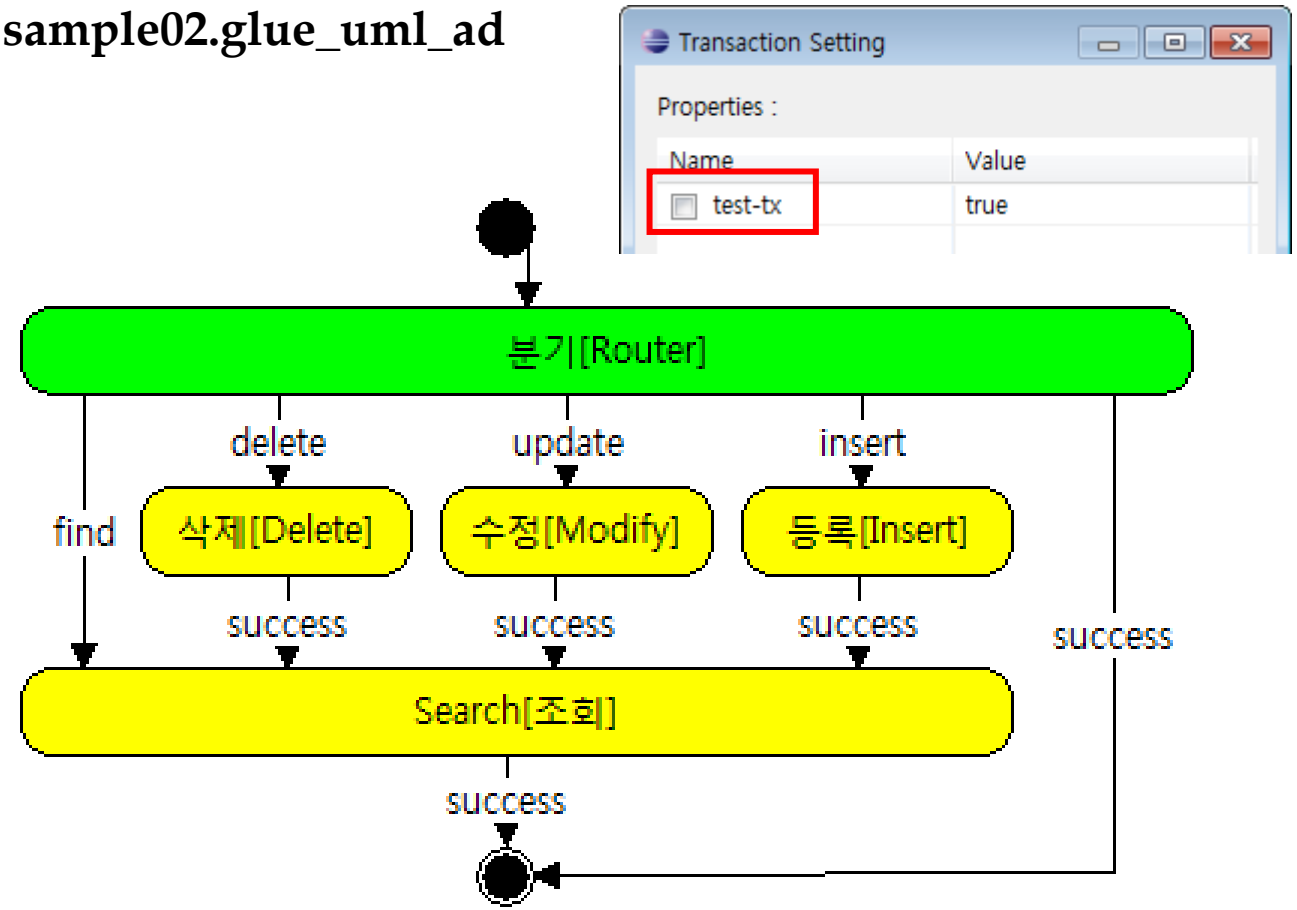
수정[Modify]

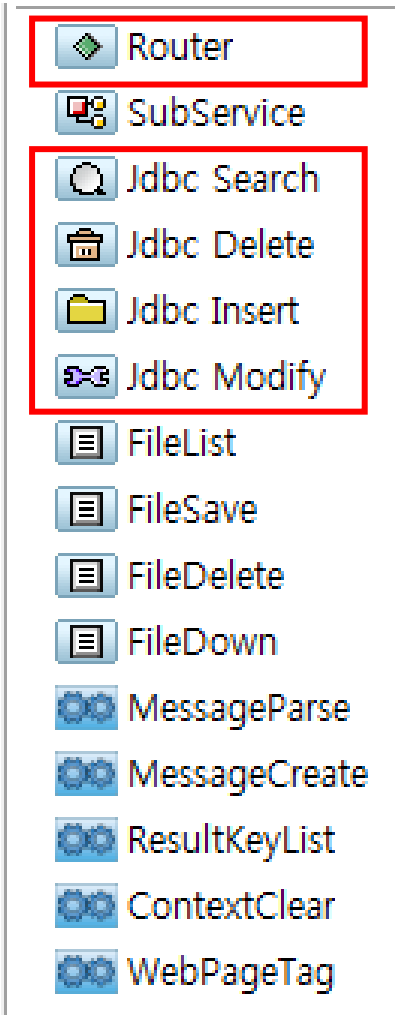
Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcModify
chk-name	chk
dao	test-dao
param-count	3
result-key	updateCnt
sql-key	sample01.emp.update
param0	SAL
param1	ENAME
param2	EMPNO

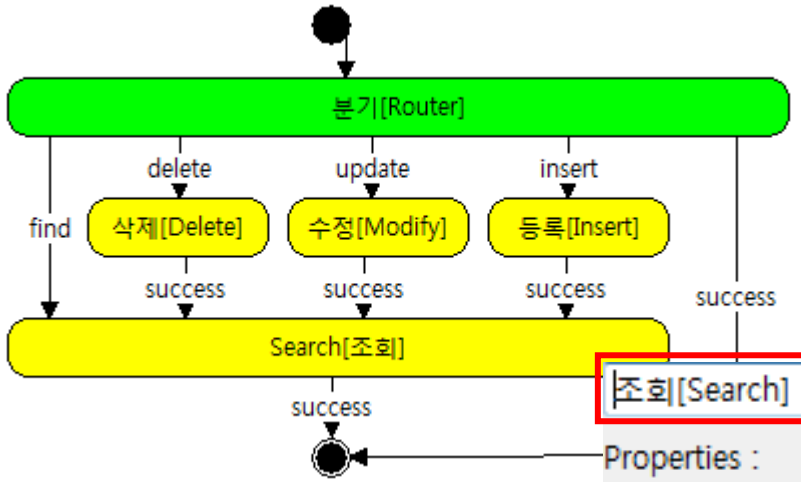
- **param-bindings 삭제**
- **list-key 삭제**
- **is-audit 삭제**

- 실습#4-3
  - sample02.glue_uml_ad





- 실습#4-3
  - sample02.glue_uml_ad



SQL

```
select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
from EMP
where DEPTNO=:deptno
```

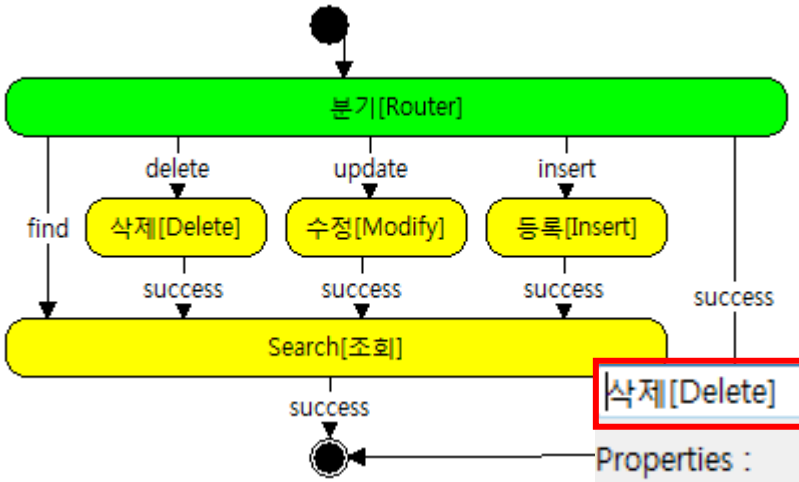
조회[Search]

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcSearch
dao	test-dao
result-key	EmpList
sql-key	sample02.emp.select

- param-count 삭제
- param0, param1, .. 삭제
- param-bindings 삭제
- cache-key 삭제
- cache-region 삭제
- cache-manager 삭제

- 실습#4-3
  - sample02.glue_uml_ad



SQL

```
delete from emp
where empno=:empno
```

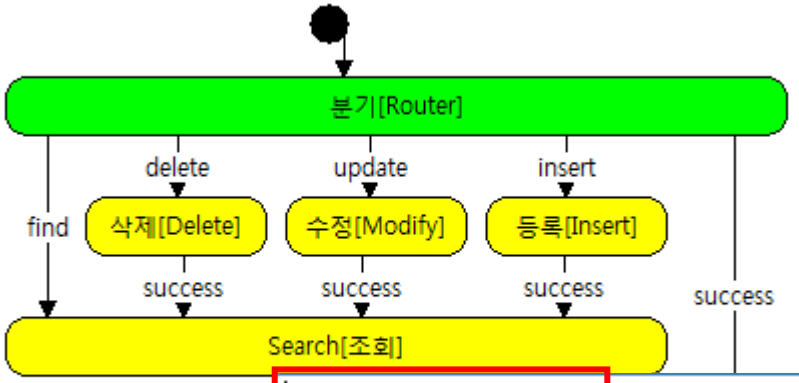
삭제[Delete]

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcDelete
chk-name	chk
dao	test-dao
result-key	deleteCnt
sql-key	sample02.emp.delete
param-bindings	empno=EMPNO

- param-count 삭제
- param0, param1, .. 삭제
- list-key 삭제

- 실습#4-3
  - sample02.glue_uml_ad



SQL

```
insert into emp(empno, ename, sal, deptno)
values(:empno, :ename, :sal, :deptno)
```

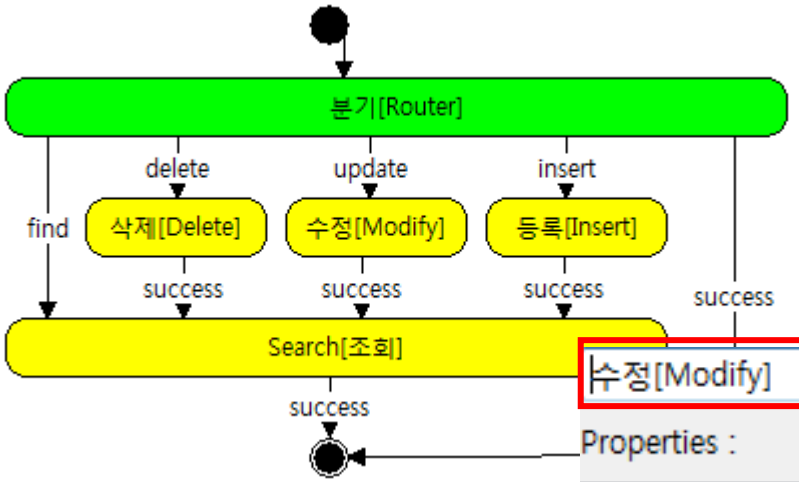
등록[Insert]

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcInsert
chk-name	chk_insert
dao	test-dao
result-key	insertCnt
sql-key	sample02.emp.insert
param-bindings	empno=empno_insert|ename=ename_insert|sal=sal_insert|deptno=deptno_insert

- param-count 삭제
- param0, param1, .. 삭제
- list-key 삭제
- is-audit 삭제

- 실습#4-3
  - sample02.glue_uml_ad



SQL

```
update emp
set sal=:sal, ename=:ename
where empno=:empno
```

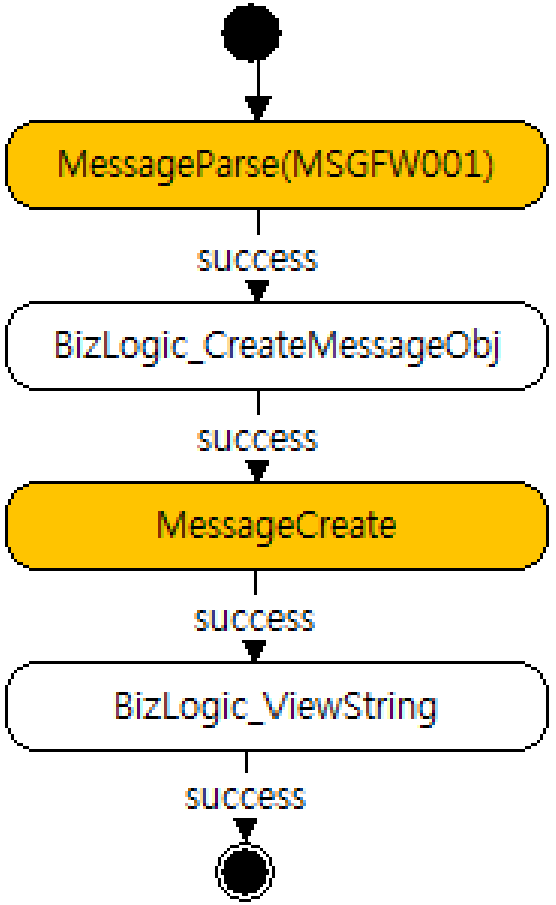
수정[Modify]

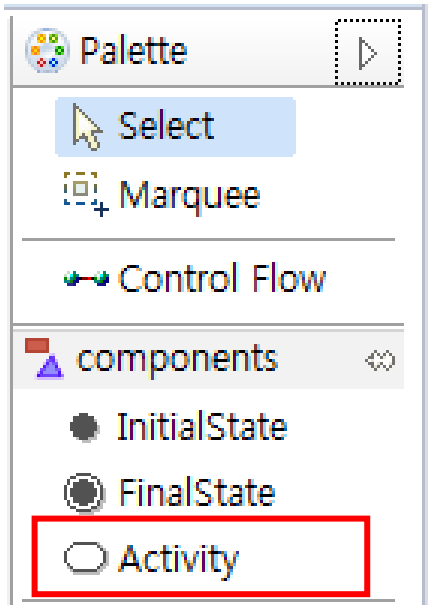
Properties :

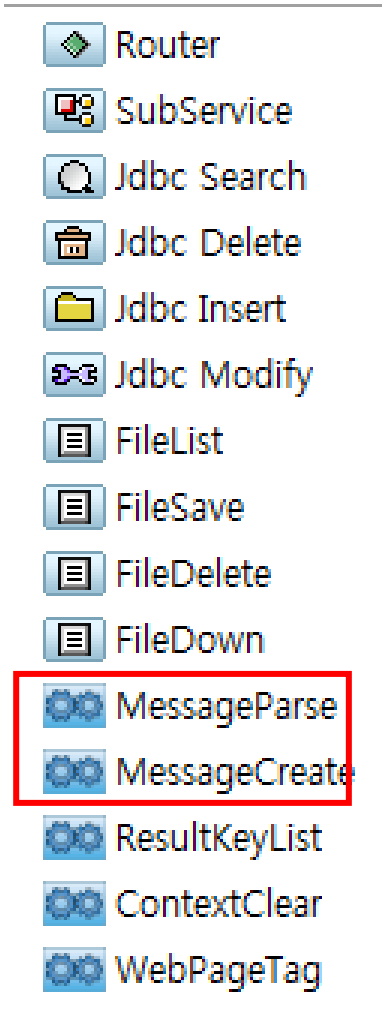
Name	Value
class	com.poscoict.glueframework.biz.activity.GlueJdbcModify
chk-name	chk
dao	test-dao
result-key	updateCnt
sql-key	sample02.emp.update
param-bindings	empno=EMPNO|sal=SAL|ename=ENAME

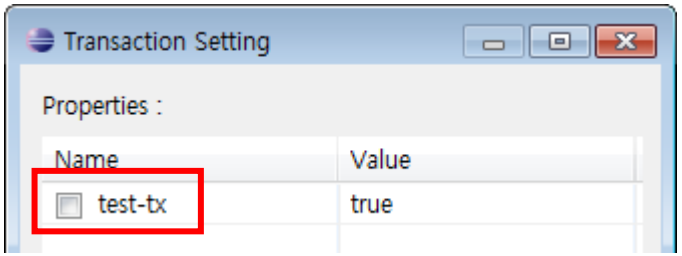
- **param-count 삭제**
- **param0, param1, .. 삭제**
- **list-key 삭제**
- **ls-audit 삭제**

- 실습#4-4
  - MSGFW001.glue_uml_ad



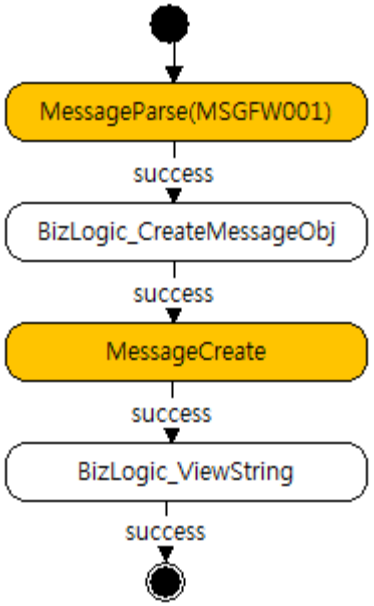






실습#4-4

MSGFW001.glue_uml_ad



MessageParse(MSGFW001)

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueMessageParse
layout	`layoutInXml`

- charset 삭제
- charset-name 삭제
- id-length 삭제
- interface-type 삭제

```
MSGFW001R50dname:abc      dloc:seoul
MSGFW001D50
MSGFW001C50dname:abc      dloc:seoul
MSGFW001U50dname:ABC      dloc:SEOUL
```

실습#4-4

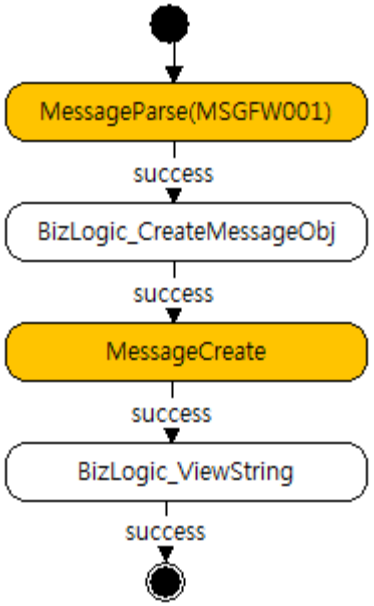
MSGFW001.glue_uml_ad

MessageCreate

Properties :

Name	Value
class	com.poscoict.glueframework.biz.activity.GlueMessageCreate
layout	`layoutInXml`
message-key	`messageObj`
result-key	`stringObj`

- 실습#4-4
  - MSGFW001.glue_uml_ad



BizLogic_CreateMessageObj

Properties :

Name	Value
class	sample.activity.GetEmp

BizLogic_ViewString

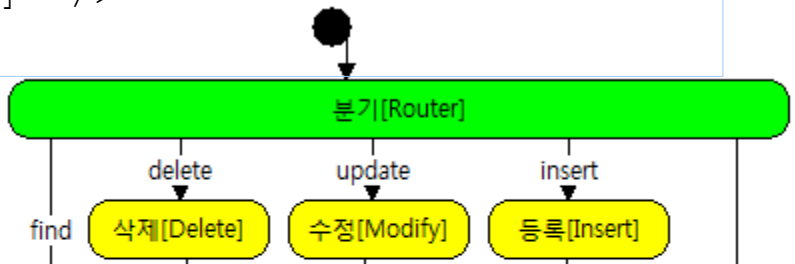
Properties :

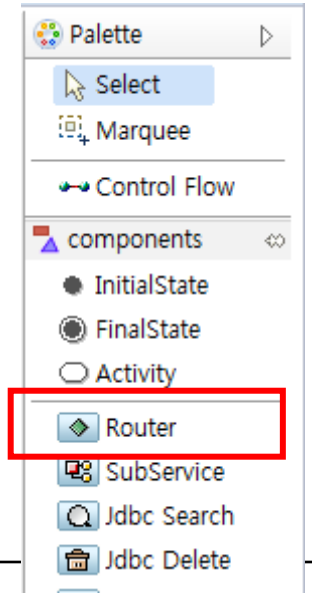
Name	Value
class	sample.activity.ViewString

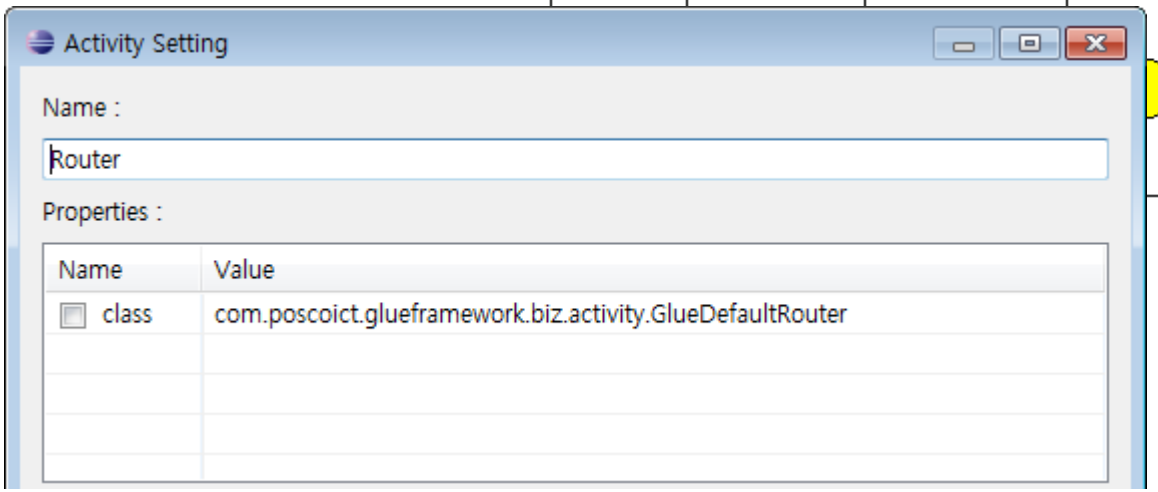
Routing Activity

```
<activity name="분기[Router]"
          class="com.poscoict.glueframework.biz.activity.GlueDefaultRouter">
    <transition name="success" value="end" />
    <transition name="find" value="조회[Search]" />
    <transition name="delete" value="삭제[Delete]" />
    <transition name="insert" value="등록[Insert]" />
    <transition name="update" value="수정[Modify]" />
</activity>
```

- 녹색
- property 없음, transition 만 있음.



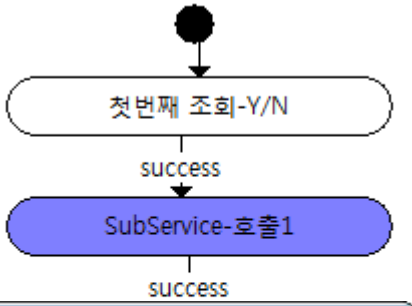


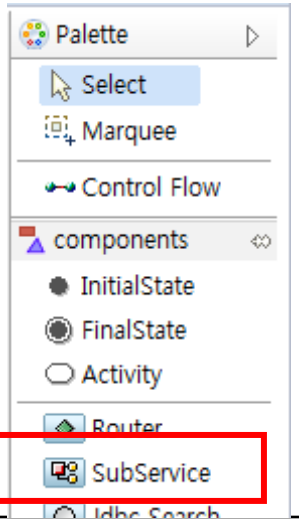


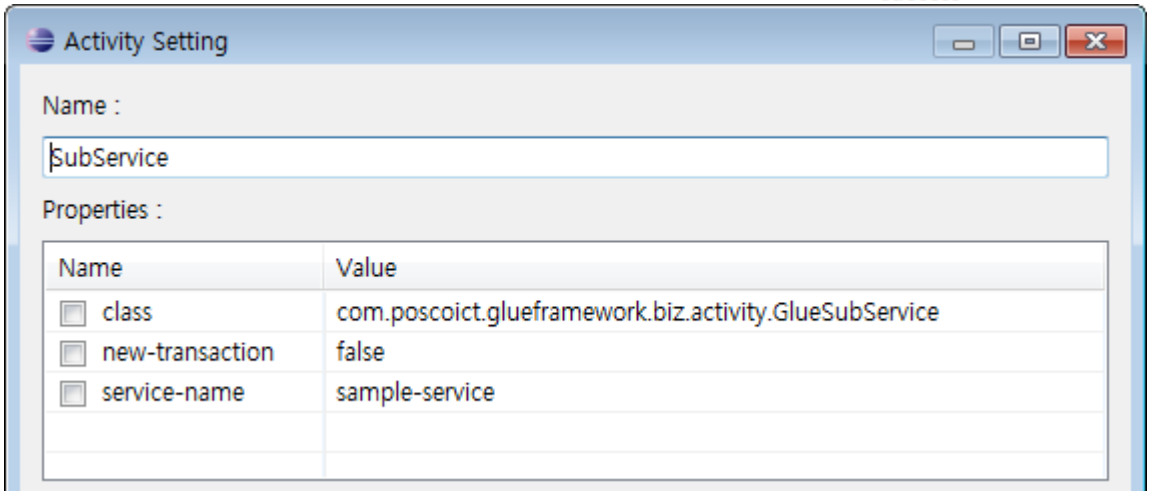
SubService Activity

```
<activity name="SubService"
          class="com.poscoict.glueframework.biz.activity.GlueSubService">
    <transition name="success" value="BizLogic" />
    <property name="service-name" value="sample03-service" />
    <property name="new-transaction" value="false" />
</activity>
```

- 푸른색
- property
  - service-name : 필수, Glue Service 명
  - new-transaction : 필수, true, false

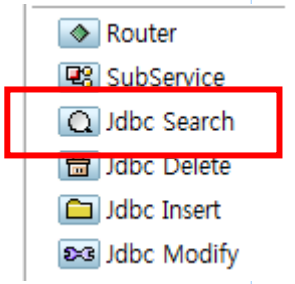






Jdbc Search Activity

```
<activity name="조회"
          class="com.poscoict.glueframework.biz.activity.GlueJdbcSearch">
    <transition name="success" value="end" />
    <property name="result-key" value="EmpList" />
    <property name="sql-key" value="sample01.emp.select" />
    <property name="dao" value="test-dao" />
    <property name="param-count" value="1" />
    <property name="param0" value="deptno" />
</activity>
```



- **노란색**
- **property**
  - dao : 필수, DAO bean id
  - sql-key : 필수, Query id
    - Query Definition의 isNamed 속성에 따라 parameter 관련 property를 선택적으로 사용해야 함.
  - result-key : Default는 <sql-key>_resultList
  - param-count , param#, param-bindings
  - cache-key, cache-region, cache-manager

- **Jdbc Search Activity**
  - **property 계속**
    - param-count : binding할 개수.
      - sql의 '?'의 개수
    - param# : binding value
      - param0, param1, param2 형태
      - ? 개수와 순서가 일치되도록
      - param-count property와 같이 사용됨
    - param-bindings : named param과 binding value
      - sqlParamName=inputName|sqlParamName=inputName 형태.
      - sqlParamName과 inputName이 같을 경우 생략.
  - **SQL 유형**

SQL

```
select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
from EMP
where DEPTNO=?
```

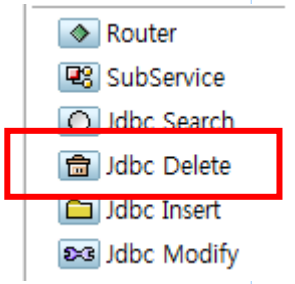
SQL

```
select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
from EMP
where DEPTNO=:deptno
```

- **Jdbc Search Activity**
  - **property 계속**
    - cache-key : 캐싱 키
      - property가 없으면 항상 dao를 이용한 조회 결과를 ctx에 담음
      - property가 있으면 CacheManager를 통해 주어진 영역에 캐싱된 Data가 있는지 판단함.
        캐싱된 Data가 없을 경우 dao를 이용한 조회 결과를 ctx에 담고
        캐싱된 Data가 있으면 캐싱된 Data를 ctx에 담음.
    - cache-region : 캐싱 영역
    - cache-manager : CacheManager bean id.
      - applicationContext.xml 의 정의한 bean id

Jdbc Delete Activity

```
<activity name="삭제"
          class="com.poscoict.glueframework.biz.activity.GlueJdbcDelete">
    <transition name="success" value="end" />
    <property name="result-key" value="deleteCnt" />
    <property name="sql-key" value="sample01.emp.delete" />
    <property name="dao" value="test-dao" />
    <property name="param-count" value="1" />
    <property name="param0" value="empno" />
</activity>
```

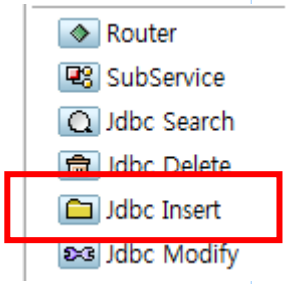


- **노란색**
- **property**
  - dao : 필수, DAO bean id
  - sql-key : 필수, Query id
    - Query Definition의 isNamed 속성에 따라 parameter 관련 property를 선택적으로 사용해야 함.
  - result-key : Default는 <sql-key>_deleteCnt
  - param-count : Jdbc Search Activity의 property 참고
  - param# : Jdbc Search Activity의 property 참고
  - param-bindings : Jdbc Search Activity의 property 참고

- **Jdbc Delete Activity**
  - **property 계속**
    - chk-name : looping data 처리용
      - Binding 값이 String[] 형태임.
      - list-key property와 같이 사용하지 못함.
    - list-key : looping data 처리용
      - Binding 값이 List<Map> 형태임
      - chk-name property와 같이 사용하지 못함.

Jdbc Insert Activity

```
<activity name="등록"
          class="com.poscoict.glueframework.biz.activity.GlueJdbcInsert">
    <transition name="success" value="end" />
    <property name="result-key" value="insertCnt" />
    <property name="sql-key" value="sample01.emp.insert" />
    <property name="dao" value="test-dao" />
    <property name="param-count" value="3" />
    <property name="param0" value="empno" />
    <property name="param1" value="ename" />
    <property name="param2" value="deptno" />
</activity>
```



- **노란색**
- **property**
  - dao : 필수, DAO bean id
  - sql-key : 필수, Query id
    - Query Definition의 isNamed 속성에 따라 parameter 관련 property를 선택적으로 사용해야 함.
  - result-key : Default는 <sql-key>_insertCnt

- **Jdbc Insert Activity**
  - **property 계속**
    - param-count : Jdbc Search Activity의 property 참고
    - param# : Jdbc Search Activity의 property 참고
    - param-bindings : Jdbc Search Activity의 property 참고
    - chk-name : Jdbc Delete Activity의 property 참고
    - list-key : Jdbc Delete Activity의 property 참고

    - is-audit : Audit 항목의 binding 값을 추가
      - glue.properties 에 audit 정보 설정 필요.
      - Glue Service 실행 전 audit data 설정 필요.

SQL

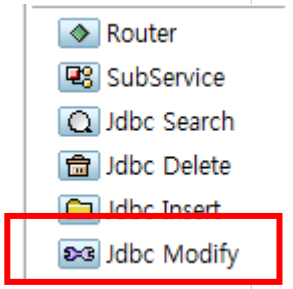
```
insert into EMP_AUDIT(
  CREATE_ID,CREATE_IP,CREATE_DT,UPDATE_ID,UPDATE_IP,UPDATE_DT,
  empno, ename, sal, deptno)
values(?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

SQL

```
insert into EMP_AUDIT(
  CREATE_ID,CREATE_IP,CREATE_DT,UPDATE_ID,UPDATE_IP,UPDATE_DT,
  empno, ename, sal,deptno)
values(:audit_id, :audit_ip, :audit_dt, :audit_id, :audit_ip, :audit_dt,
  :empno, :ename, :sal, :deptno)
```

Jdbc Modify Activity

```
<activity name="수정"
          class="com.poscoict.glueframework.biz.activity.GlueJdbcModify">
    <transition name="success" value="end" />
    <property name="result-key" value="updateCnt" />
    <property name="sql-key" value="sample01.emp.update" />
    <property name="dao" value="test-dao" />
    <property name="param-count" value="3" />
    <property name="param0" value="sal" />
    <property name="param1" value="ename" />
    <property name="param2" value="empno" />
</activity>
```



- **노란색**
- **property**
  - dao : 필수, DAO bean id
  - sql-key : 필수, Query id
    - Query Definition의 isNamed 속성에 따라 parameter 관련 property를 선택적으로 사용해야 함.
  - result-key : Default는 <sql-key>_updateCnt

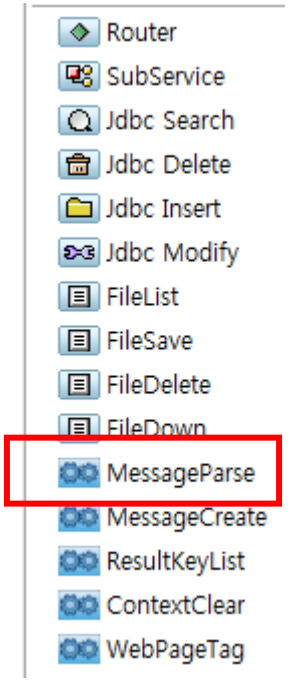
- **Jdbc Modify Activity**
  - **property 계속**
    - param-count : Jdbc Search Activity의 property 참고
    - param# : Jdbc Search Activity의 property 참고
    - param-bindings : Jdbc Search Activity의 property 참고
    - chk-name : Jdbc Delete Activity의 property 참고
    - list-key : Jdbc Delete Activity의 property 참고

    - is-audit : Jdbc Insert Activity의 property 참고

MessageParse Activity

```
<activity name="메세지파싱"
          class="com.poscoict.glueframework.biz.activity.GlueMessageParse">
    <transition name="success" value="BizLogic" />
    <property name="layout" value="layoutInDB" />
</activity>
```

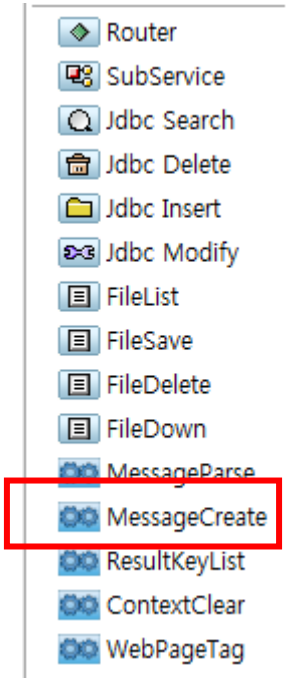
- 주황색
- property
  - layout : 필수, Layout Manager bean id
  - message-id :
  - id-length : Default는 8
    - 메시지 ID 길이
    - 수신 Data의 substring 기준.
  - charset : 고정 값
    - EUC-KR 등
  - charset-name : charset 참조 값(ctx key)
    - Glue Service 시작 전에 외부 설정 값을 읽어오기 위한 Key
- **인코딩 우선순위 : charset-name → charset → system charset**



MessageCreate Activity

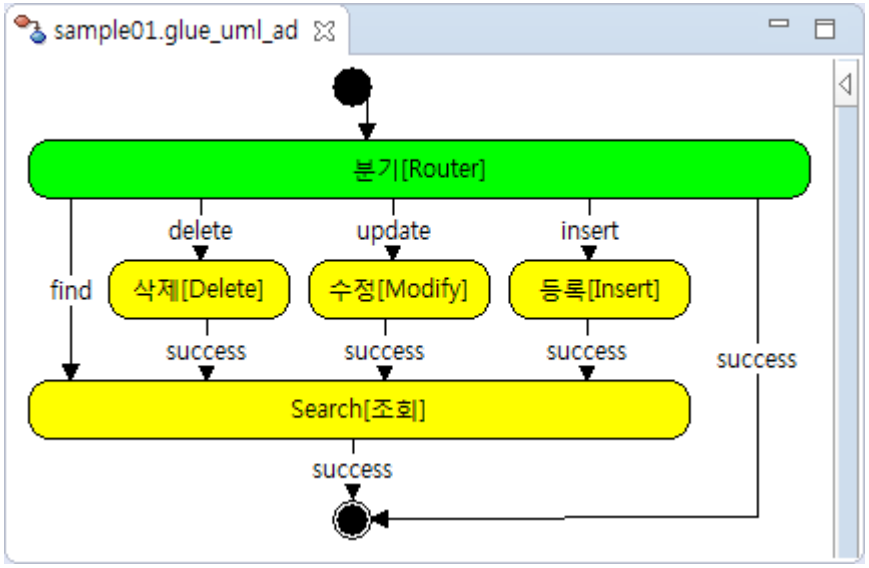
```
<activity name="메시지생성"
          class="com.poscoict.glueframework.biz.activity.GlueMessageCreate">
    <transition name="success" value="BizLogic" />
    <property name="layout" value="layoutInDB" />
    <property name="message-key" value="messageObj" />
</activity>
```

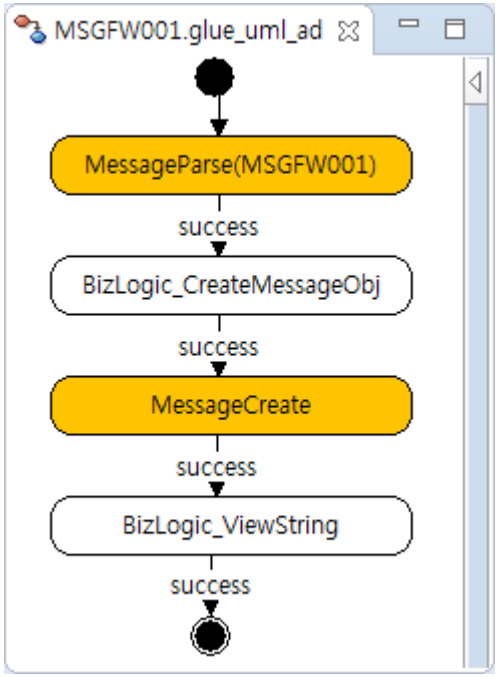
- **주황색**
- **property**
  - layout : 필수, Layout Manager bean id
  - message-key : 필수
    - GlueMessage Object
  - result-key : Default는 <activity name>_result
  - charset
  - charset-name



Glue Framework 제작 가이드

Service 란?





- Service는 독립적인 Business의 한 기능을 수행하는 단위로 보통 재사용이 가능한 단위로 작성하게 된다. 즉 Service는 아주 작은 하나의 기능단위로 할 수도 있고 반대로 모든 Business를 하나의 Service로도 처리할 수 있지만 이것은 설계자의 능력에 따라 정의 될 수 있다.

- **Serivce 파일**
  - **Glue Activity Diagram 으로부터 생성된 파일**
  - **transaction-manager와 activity로 구성되어 있음.**
  - **Service 파일 이름 규칙**
    - ServiceName.glue_uml_ad ➔ ServiceName-serivce.xml
  - **애플리케이션 상의 위치**
    - service\
      - GlueSample.war! \WEB-INF \ classes \ service
      - user-application.jar! \ service
  - **Glue Project 상의 위치**
    - src\service
      - <sample> \ src \ service
      - <sample-maven> \ src \ main \ resource \ service

- **Service**
  - **name 속성 : file명과 일치함**
  - **initial 속성**
  - **transaction-manager 요소 : 0개 이상**
    - id 속성
    - commit 속성
  - **activity 요소 : 1개 이상**
    - name 속성
    - class 속성
    - transition 요소 : 1개 이상
    - property 요소 : 0개 이상

```
<?xml version="1.0" encoding="UTF-8"?>
<service name="sample01-service" initial="Router" >
    <transaction-manager id="test-tx" commit="true"/>
    <activity name="SaveData"
              class="sample.activity.SaveData">
        <transition name="success" value="NextActivityName"/>
        <property name="dao" value="test-dao"/>
        <property name="result-key" value="saveCnt"/>
    </activity>
    . . .
```

Activity 구현

GlueActivity 상속 및 runActivity() 구현

```
public class CustomActivity extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        . . .
        return GlueBizControlConstants.SUCCESS;
    }
}
```

GlueContext 이용 : get(), put()

```
String[] deptno = (String[])ctx.get("deptno");
String[] empno = (String[])ctx.get("empno");

ctx.put("DelCount", obj );
ctx.put("UsrMsg", n+"건이 정상적으로 처리되었습니다.");
```

- getMessage(), setMessage() : GlueMessage 용
- getException(), setException() : GlueException 용
- getMultiPartRequest() : GlueMultipartRequest 용

Activity 구현

종료 처리 : return

```
<activity name="분기" class="sample.activity.CustomRouter">
    <transition name="success" value="추가" />
    <transition name="add" value="추가" />
    <transition name="remove" value="삭제" />
    <transition name="modify" value="수정" />
    <transition name="send" value="전송" />
</activity>
```

transition 을 Source Code(Activity)에서 다음과 같이 이용

```
if( ... ){
    return "add";
}else if( ... ) {
    return "remove";
}else if( ... ) {
    return "modify";
}else if( ... ) {
    return "delete";
}
return GlueBizControlConstants.SUCCESS;
```

Activity 구현

Property 이용 : getProperty()

```
<activity name="삭제" class="sample.activity.CustomDelete">
    <transition name="success" value="end" />
    <property name="result-key" value="DelCount" />
    <property name="sql-key" value="emp.delete" />
    <property name="dao" value="test-dao" />
    <property name="check-name" value="chk" />
    <property name="param-count" value="2" />
    <property name="param0" value="deptno" />
    <property name="param2" value="empno" />
</activity>
```

해당 Property를 Source Code(Activity)에서 다음과 같이 이용

```
String daoName = this.getProperty("dao");
String paramCnt = this.getProperty("param-count");
```

- getPropertyNames()
- dynamicProperties

- DB Data Handling
  - DAO 종류
    - Jdbc Dao
    - MyBatis Dao
    - Hybernamte Dao
  - DAO 메소드 종류
    - 메소드의 상세한 사용 방법은 java doc을 참조

Method	return type	Description
find	List	조회/수정/삽입/삭제
update	int	
insert	int	
delete	int	

DB Data Handling

DAO 이용 : getDao()

```
public class CustomActivity extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        GlueGenericDao dao = this.getDao("test-dao");
        List rowSet = dao.find(query_id);
        . . .
        return GlueBizControlConstants.SUCCESS;
    }
}
```

Query 실행

```
List rowSet = dao.find(queryId);
int cnt = dao.update(queryId, . . .);
```

DB Data Handling

Parameter 생성 및 설정

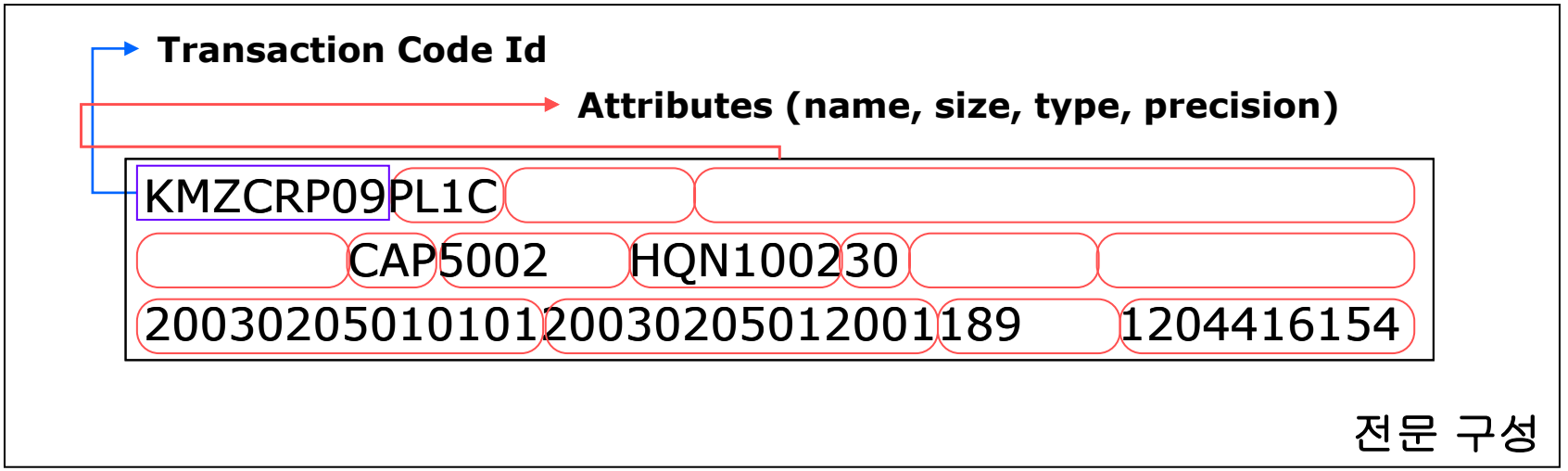
```
String [] deptno = (String[])ctx.get("DeptnoP");
String [] sal = (String[])ctx.get("SalP");

// SQL Type#1 : select * from EMP where deptno=? and sal>?
List paramValueList = new ArrayList();
paramValueList.add(deptno[0]);
paramValueList.add(sal[0]);

GlueParameter<List> parameter1 = new GlueParameter<List>();
parameter1.setParameter(paramValueList);

// SQL Type#2 : select * from EMP where deptno=:deptno and sal>:sal
Map paramValueMap = new HashMap();
paramValueMap.put("deptno", deptno[0]);
paramValueMap.put("sal", sal[0]);

GlueParameter<Map> parameter2 = new GlueParameter<Map>();
parameter2.setParameter(paramValueMap);
```

DB Data Handling

Transaction Manager 이용 : commitTransaction(), rollbackTransaction()

```
public class CustomActivity extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        . . .

        //service.xml 에 기술된 것을 사용함
        //<transaction-manager id="test-tx" commit="true"/>
        this.commitTransaction( "test-tx" );
        . . .
        return GlueBizControlConstants.SUCCESS;
    }
}
```

- **Message(TC) Handling**
  - **Layout Manager 종류**
    - XML Layout Manager
    - DB Layout Manager
  - **관련 Reuse Activity**
    - MessageParse : String 을 GlueMessage객체로
    - MessageCreate : GlueMessage객체를 String으로
  - **TC 예제**



Message(TC) Handling

GlueMessage객체 생성 예제

```
GlueMessage messageObj = new GlueMESMessageImpl();

messageObj.setTCID("MSGSEND1");

messageObj.setObject("TRANSACTION_CODE", "MSGSEND1");
messageObj.setObject("DEPTNO", "50");
messageObj.setObject("DNAME", "SW Test Group");
messageObj.setObject("LOC", "Seoul,Korea");

List empnoList = new ArrayList();
empnoList.add("1111");
empnoList.add("2222");
messageObj.setObject("EMPNO", empnoList);

List enameList = new ArrayList();
enameList.add("jane park");
enameList.add("john park");
messageObj.setObject("ENAME", enameList);
```

Cache Data Handling

Cache 정의 및 목적

- 자바 오브젝트를 저장하는 메모리 상의 공간
- 오브젝트 생성에 비용이 많이 소모되는 경우나 빈번하게 사용되는 Read-only 성의 데이타들을 Cache에 저장하여 사용함으로 성능향상을 꾀한다.

Cache Manager 종류

- EhCache Cache Manager
- JCS Cache Manager

Cache Manager 메소드 종류

- 메소드의 상세한 사용 방법은 java doc을 참조

Method	return type	Description
getCacheObject	Object	조회/등록또는갱신/삭제/전체 삭제
putCacheObject		
removeCacheObject		
clear		
isPresent	boolean	

Cache Data Handling

CacheManager 사용 예제

```
GlueCacheManager cacheManager
   = (GlueCacheManager)GlueStaticContext.getBeanFactory().getBeanObject("cacheManager");

Object cacheKey = "dept.list";
List deptList = null;
if(cacheManager.isPresent(cacheKey)){
    deptList = (List)cacheManager.getCacheObject(cacheKey);
}else{
    deptList = new ArrayList();
    deptList.add(new String[]{"10","Human Resources"});
    deptList.add(new String[]{"20","Development Team"});
    deptList.add(new String[]{"30","Test Team"});
    cacheManager.putCacheObject(cacheKey, deptList);
}
```

- **Logging**
  - **Logging 정의 및 역할**
    - 어떤 원인 때문에 에러가 발생했는지에 대한 세부 정보를 제공
    - 체계적이고 상세한 logging 메시지는 개발 시점뿐만 아니라 운영하는 시점에도 상당한 위력을 발휘함.
    - SLF4J 를 통해 Logging System 변경 가능.
  - **Logging system**
    - logback
    - log4j
    - Jakarta common logging
    - . . .

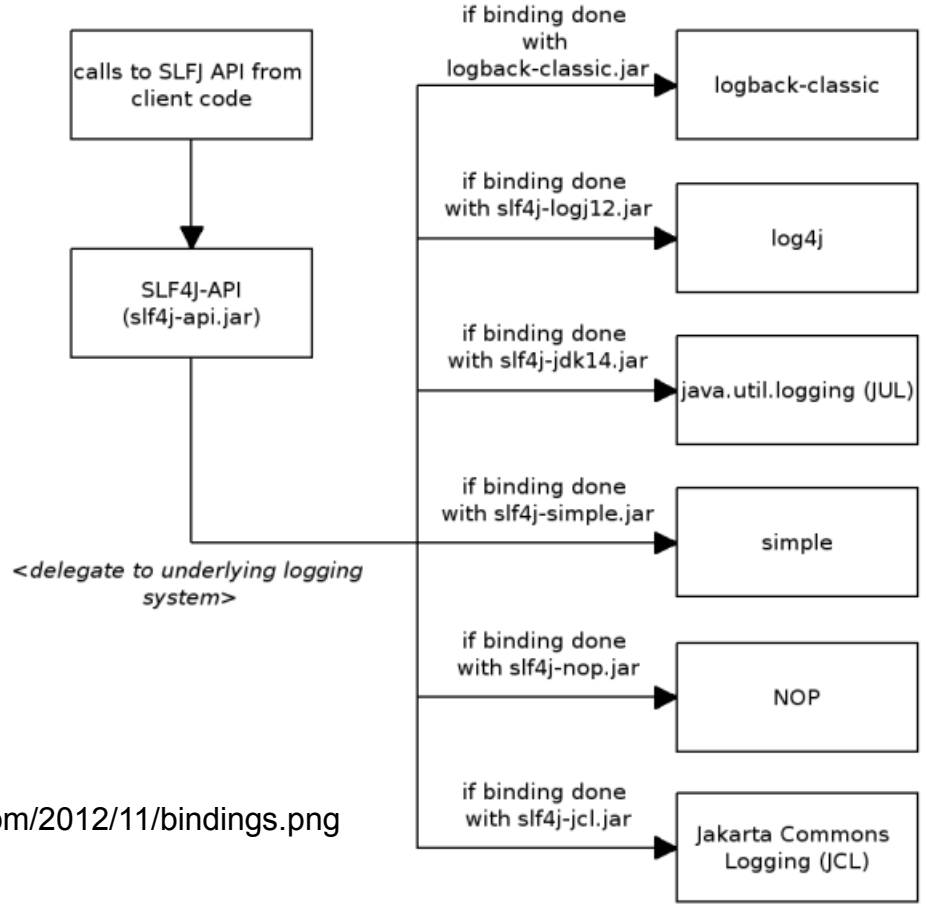


그림 출처 : http://beyondj2ee.files.wordpress.com/2012/11/bindings.png

- Logging
  - GlueLog 메소드 종류
    - 메소드의 상세한 사용 방법은 java doc을 참조

Method	Description
trace	Level : trace > debug > info > warn > error
debug	
info	
warn	
error	

Logging

Logger 사용예 – Activity

```
public class LogTestActivity extends GlueActivity<GlueContext> {
    public String runActivity(GlueContext ctx) {
        // 현재 로그 레벨이 디버그 이상이면
        // (로그레벨, trace < debug < info < warn < error )
        if(this.logger.isDebugEnabled(){
            this.logger.debug("debug logging test");
        }
        logger.info("info logging test");
        if( 조건체크 ){
            try{
                // Biz Logic
            }catch(Exception e){
                // 에러 메시지와 Exception Stacktrace를 로깅한다.
                logger.error("error logging test : " + e.getMessage(), e);
            }
        }else{
            logger.warn("warn logging test");
        }
        return GlueBizControlConstants.SUCCESS;
    }
}
```

Logging

Logger 사용예 – Activity 외

```
public class LoggingClient {
    // 각 클래스별로 GlueLog 를 등록함
    protected GlueLog logger = GlueLogFactory.getLogger(LoggingClient.class);

    public void performTask() {
        try {
            // 현재 로그 레벨이 디버그 이상이면
            // (로그레벨, trace < debug < info < warn < error )
            if (logger.isDebugEnabled()) {
                // debug logging
                logger.debug("debug logging test");
            }
            // 로깅할 메시지가 info 레벨 이상이 되는 경우는
            // 현재 로그 레벨을 체크하지 않아도 상관없다.
            logger.info("info logging test");
        } catch (GlueException ex) {
            // 에러 메시지와 Exception Stacktrace를 로깅한다.
            logger.error("Fail to perform task: " + ex.getMessage(), ex);
        }
    }
}
```

Exception 처리

Glue Framework 에서의 Exception 처리 원칙

- 다음의 경우를 제외하고 모든 Exception을 Unchecked Exception을 사용한다.
  - 회복 가능한 에러가 발생한 경우 (시간이 지나면 복구되는 에러)
  - 모든 호출자들이 발생 되는 Exception을 꼭 처리해야 하는 경우
    - 비즈니스적인 Exception
- GlueException(Unchecked Exception)을 확장하여 정의한다.
  - GlueDaoException, GlueCacheException, GlueMessageParserException 등등..

Checked Exception 의 단점

- 소스코드의 복잡도 증가
- 소스코드의 가독성(Readability) 저하
  - 유지보수의 어려움
- 비즈니스적인 Exception을 제외하고는 개발자가 프로그램적으로 처리할 수 있는 것이 상당히 제한적이다. Logging 정의 및 역할

Exception 처리

Checked Exception

- java.lang.Exception을 상속한 Exception으로 try~catch 구문으로 처리하지 않을 경우 컴파일 에러가 발생한다. 그러므로, 컴파일 시점에 Exception 처리가 제대로 구현되어 있는지를 확인이 가능하다. 예를 들어 java.io.IOException이 대표적인 Checked Exception이다.

Unchecked Exception

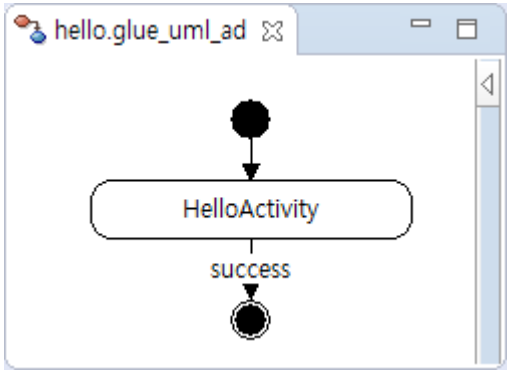
- java.lang.RuntimeException을 상속한 Exception으로 try~catch 구문으로 처리하지 않아도 컴파일 에러가 발생하지 않는다. 단, 런타임 시점에 에러가 발생할 경우 Exception이 발생한다. 예를 들어, java.lang.NullPointerException이 대표적인 Unchecked Exception이다.

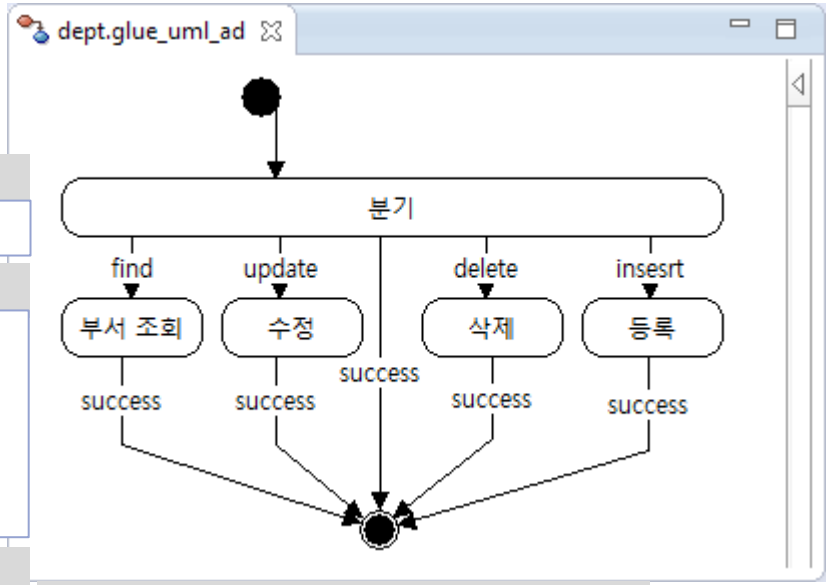
Checked Exception	Unchecked Exception

`public Task getTask() throws Exception;`
 	
`public void performTask() {`
`  try {`
`  // … pre processing`
`  Task task = getTask();`
`  task.start();`
`  // … post processing`
`  } catch (Exception e) {`
`  // handle exception`
`  }`
`}`
	`public Task getTask() throws RuntimeException;`
	
	`public void performTask() {`
	`  Task task = getTask();`
	`  task.start();`
	`}`

- 실습#5
  - **sample.glue_uml_ad**
    - sample.activity.SearchEmp
  - **sample01.glue_uml_ad**
  - **sample02.glue_uml_ad**
  - **MSGFW001.glue_uml_ad**
    - sample.activity.GetEmp (실습#5-1)
    - sample.activity.ViewString (실습#5-2)

- 실습#5
  - **hello.glue_uml_ad**
    - sample.activity.HelloActivity (실습#5-3)
  - **dept.glue_uml_ad**
    - sample.activity.DeptRouter (실습#5-4)
    - sample.activity.DeptSearch (실습#5-5)
    - sample.activity.DeptDelete (실습#5-6)
    - sample.activity.DeptUpsert (실습#5-7)
    - test-tx, test-dao





분기[sample.activity.DeptRouter]

ctx-keys	DeptList|cnt

수정[sample.activity.DeptUpsert]

dao	test-dao
dml-type	update
update-loop	chk
update-sql	dept.update
update-params	DNAME,LOC,DEPTNO
result-key	cnt

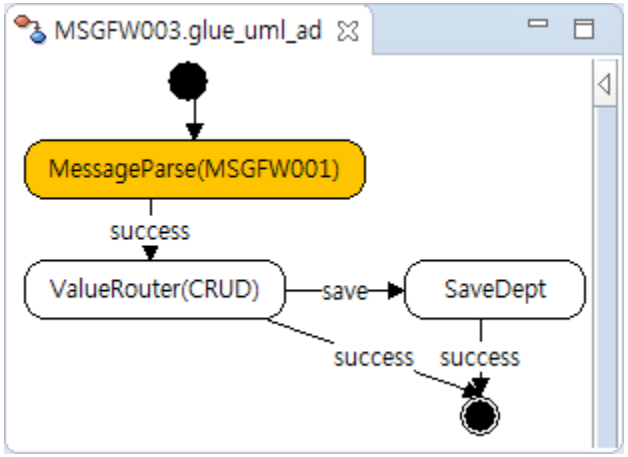
등록[sample.activity.DeptUpsert]

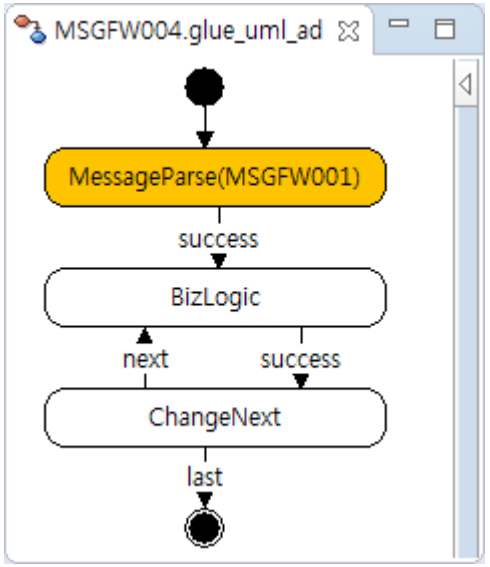
dao	test-dao
dml-type	insert
insert-sql	dept.insert
insert-params	DEPTNO,DNAME,LOC
result-key	cnt

삭제[sample.activity.DeptDelete]

dao	test-dao
select-sql	sample.emp.select
delete-sql	dept.delete
param-key	DEPTNO

- 실습#5
  - **MSGFW003.glue_uml_ad**
    - sample.activity.ValueRouter
    - sample.activity.SaveDept
    - test-tx, test-dao
  - **MSGFW004.glue_uml_ad**
    - sample.activity.BizLogic
    - sample.activity.ChangeNext
    - dept-service



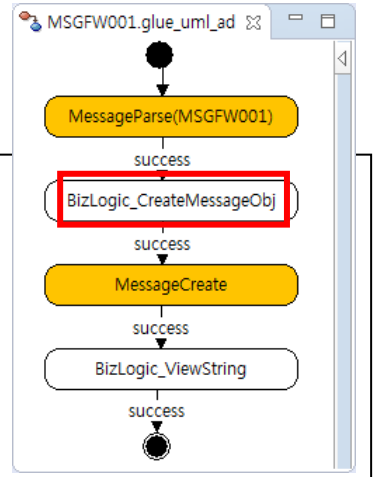


MessageParse(MSGFW001)	
layout	layoutInXml
message-id	MSGFW001

SaveDept[sample.activity.SaveDept]	
dao	test-dao
insert-sql	dept.insert
insert-param	DEPTNO,DNAME,LOC
update-sql	dept.update
update-param	DNAME,LOC,DEPTNO
delete-sql	dept.delete
delete-param	deptno=DEPTNO

실습#5-1

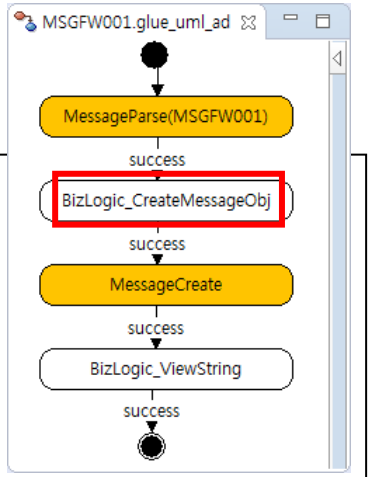
sample.activity.GetEmp (1/3)




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.message.*;
public class GetEmp extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        GlueMessage message = ctx.getMessage();

        GlueMessage messageObj = new GlueMESMessageImpl();
        messageObj.setTCID( "MSGFW002" );
        messageObj.setObject( "TRANSACTION_CODE", "MSGFW002" );
        messageObj.setObject( "CRUD", message.get( "CRUD" ) );
        messageObj.setObject( "DEPTNO", message.get( "DEPTNO" ) );
        messageObj.setObject( "DNAME", message.get( "DNAME" ) );
        messageObj.setObject( "LOC", message.get( "DLOC" ) );

        List<Object> empno = new ArrayList<Object>();
        empno.add( 1101 );
        empno.add( 1102 );
        empno.add( 1103 );

        messageObj.setObject( "EMPNO", empno );
```

실습#5-1

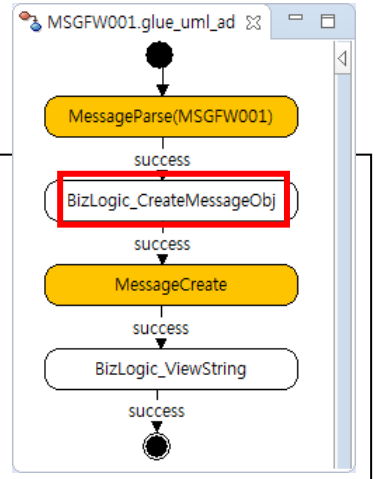
sample.activity.GetEmp (2/3)




```
        List<Object> ename = new ArrayList<Object>();
        ename.add( "name#1" );
        ename.add( "name#2" );
        ename.add( "name#3" );

        messageObj.setObject( "ENAME", ename );
        ctx.put( "messageObj", messageObj );

/*
        List<GlueMessage> messageList = new ArrayList<GlueMessage>();
        GlueGenericDao dao = this.getDao( "test-dao" );
        List<Object> paramList = new ArrayList<Object>();
        paramList.add( message.get( "DEPTNO" ) );
        GlueParameter<List<Object>> param = new GlueParameter<List<Object>>();
        param.setParameter( paramList );
        List<Map<String, Object>> rowSet = dao.find( "sample01.emp.select" , param );
        for(int cnt=rowSet.size(), i=0; i<cnt; ){
            GlueMessage msg = new GlueMESSMessageImpl();
            msg.setTCID( "MSGFW002" );
            msg.setObject( "TRANSACTION_CODE", "MSGFW002" );
            msg.setObject( "CRUD", message.get( "CRUD" ) );
            msg.setObject( "DEPTNO", message.get( "DEPTNO" ) );
            msg.setObject( "DNAME", message.get( "DNAME" ) );
            msg.setObject( "LOC", message.get( "DLOC" ) );
```

실습#5-1

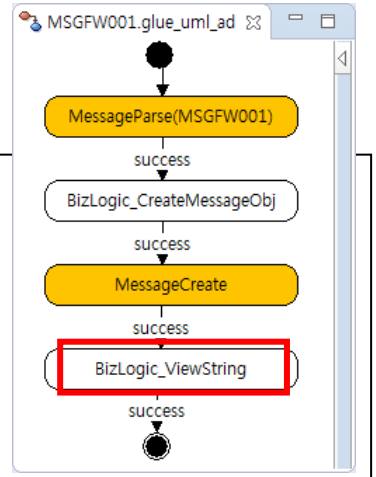
sample.activity.GetEmp (3/3)




```
            List<Object> empnoList = new ArrayList<Object>();
            List<Object> enameList = new ArrayList<Object>();
            for(int j=0; j<3; j++, i++){
                if(i<cnt){
                    Map<String, Object> row = rowSet.get( i );
                    empnoList.add( row.get( "EMPNO" ) );
                    enameList.add( row.get( "ENAME" ) );
                }else{
                    empnoList.add( "" );
                    enameList.add( "" );
                }
            }
            msg.setObject( "EMPNO", empnoList );
            msg.setObject( "ENAME", enameList );
            messageList.add( msg );
        }
        ctx.put( "messageObj", messageList );
*/
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-2

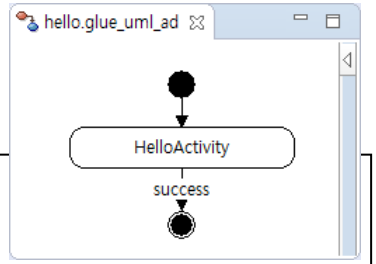
sample.activity.ViewString




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
public class ViewString extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        Object send = ctx.get( "stringObj" );

        if ( send instanceof Map<?, ?> ) {
            Map<String, List<String>> sendMap = (Map<String, List<String>>) send;
            for ( String key : sendMap.keySet() ) {
                List<String> sendList = sendMap.get( key );
                for ( String tc : sendList ) {
                    System.out.println( key + "["+tc+"]" );
                }
            }
        }
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-3

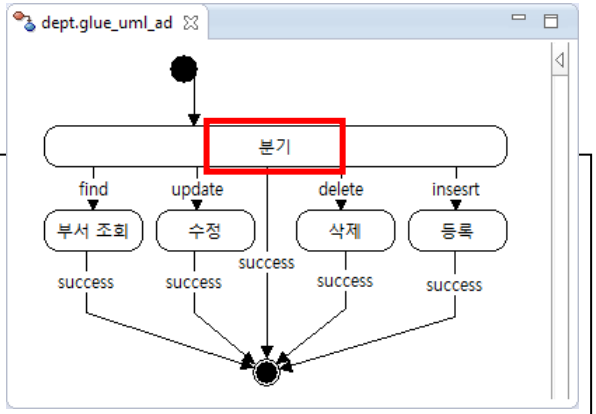
sample.activity.HelloActivity




```
package sample.activity;

import com.poscoict.glueframework.biz.activity.GlueActivity;
import com.poscoict.glueframework.biz.control.GlueBizControlConstants;
import com.poscoict.glueframework.context.GlueContext;

public class HelloActivity extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        System.out.println( "ServiceName : " +
                            ctx.get( GlueBizControlConstants.SERVICE_NAME ) );
        System.out.println( "This is '" + this.getName() + "' activity" );

        Object input = ctx.get( "input" );
        System.out.println( "data : " + input );

        Object output = "Hello " + input + "!!!";
        ctx.put( "result", output );

        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-4

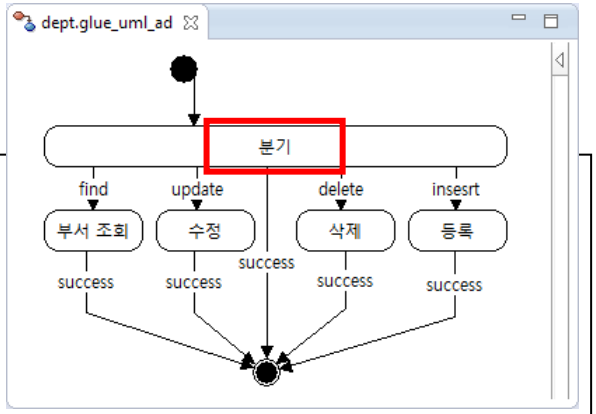
sample.activity.DeptRouter (1/2)




```
package sample.activity;

import java.util.*;

import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;

public class DeptRouter extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        /* ResultKey List (reuse activity) */
        String resultKeyList
            = this.getProperty( GlueActivityConstants.CTX_KEY_LIST );
        if ( resultKeyList != null ) {
            String[] keys = resultKeyList.split( "\\|" );
            List<String> keyList = new ArrayList<String>( keys.length );
            for ( int i = 0, iz = keys.length; i < iz; i++ ) {
                keyList.add( keys[i] );
            }
            ctx.put( GlueBizControlConstants.RESULT_KEY_LIST, keyList );
        }
```

실습#5-4

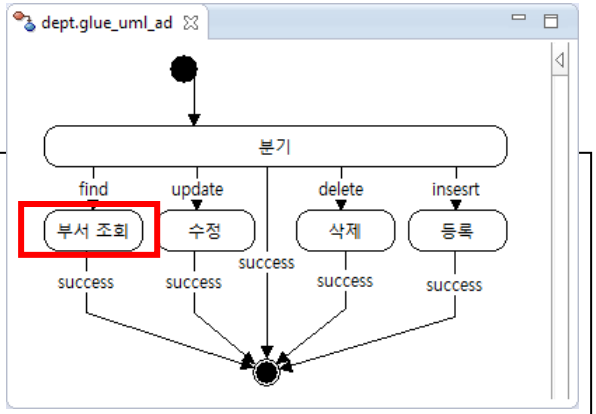
sample.activity.DeptRouter (2/2)




```
        /* Default Router : Key 유무에 의한 분기 */
        if ( ctx.get( "find" )!=null )
            return "find";
        if ( ctx.get( "update" )!=null )
            return "update";
        if ( ctx.get( "delete" )!=null )
            return "delete";
        if ( ctx.get( "insert" )!=null )
            return "insert";
        /* Value Router : Key 값에 따라 분기 */
        if ( "U".equals(ctx.get( "CRUD" )) )
            return "update";
        if ( "D".equals(ctx.get( "CRUD" )) )
            return "delete";
        if ( "C".equals(ctx.get( "CRUD" )) )
            return "insert";
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-5

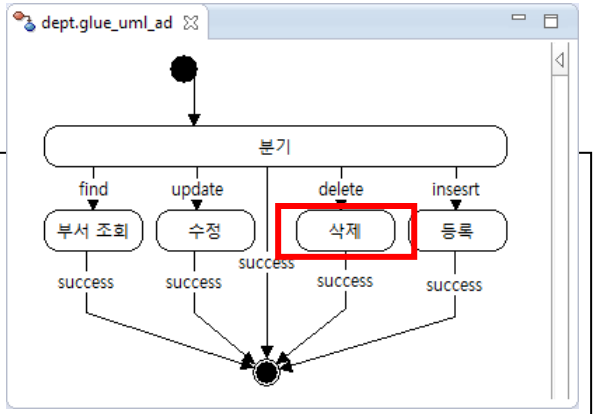
sample.activity.DeptSearch




```
package sample.activity;

import java.util.*;

import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.dao.*;

public class DeptSearch extends GlueActivity<GlueContext> {
    public String runActivity(GlueContext ctx) {
        GlueGenericDao dao = this.getDao("test-dao");

        List<?> result = dao.find("dept.select");
        ctx.put("DeptList", result);

        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-6

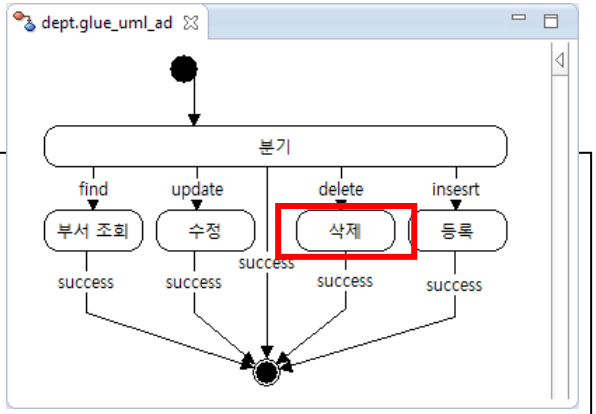
sample.activity.DeptDelete (1/2)




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.dao.*;
import com.poscoict.glueframework.dao.vo.*;
public class DeptDelete extends GlueActivity<GlueContext> {
    public String runActivity(GlueContext ctx) {
        GlueGenericDao dao = this.getDao( this.getProperty( "dao" ) );
        Object data = ctx.get( this.getProperty( "param-key" ) );
        Object value = data;
        if ( data instanceof Object[] ){
            value = ( (Object[])data )[0];
        }
        /* select EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, DEPTNO
           from EMP
           where DEPTNO=? */
        List<Object> paramList = new ArrayList<Object>();
        paramList.add( value );
        String selectKey = this.getProperty( "select-sql" );
        List<?> results
            = dao.find( selectKey, new GlueParameter<List<Object>>( paramList ) );
```

실습#5-6

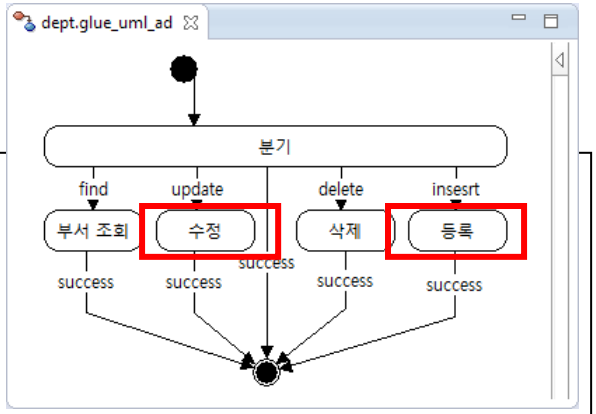
sample.activity.DeptDelete (2/2)




```
        if ( results != null && results.size() > 0 ){
            throw new GlueException( "부서원이 존재하므로" +
                                     "부서정보를 삭제할 수 없습니다." );
            /* 또는 cascading 삭제로직 수행 가능
               delete from emp where depno=? */
        }
        System.out.println( "  부서를 삭제합니다. " );
        /* delete from DEPT
           where DEPTNO=:deptno */
        Map<String, Object> paramMap
            = new HashMap<String, Object>();
        paramMap.put( "deptno", value );
        String deleteKey = this.getProperty( "delete-sql" );
        int cnt
            = dao.delete( deleteKey, new GlueParameter<Map<String,Object>>( paramMap ) );
        ctx.put( "cnt", cnt );
        System.out.println( " delete count : " + cnt );

        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-7

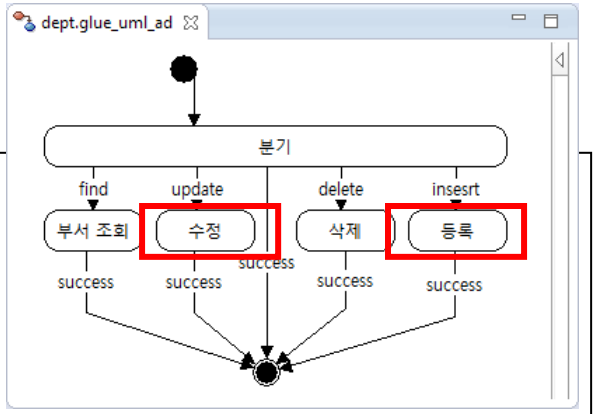
sample.activity.DeptUpsert (1/3)




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.dao.*;
import com.poscoict.glueframework.dao.vo.*;
public class DeptUpsert extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        GlueGenericDao dao = this.getDao( this.getProperty( "dao" ) );
        String type = this.getProperty( "dml-type" );
        int cnt = 0;
        if ( "insert".equals( type ) ) {
            String[] paramNames = this.getProperty( "insert-params" ).split( "," );
            List<Object> paramList = new ArrayList<Object>();
            for ( String name : paramNames ){
                Object value = ctx.get( name );
                if ( value instanceof Object[] )
                    paramList.add( ( (Object[])value )[0] );
                else
                    paramList.add( value );
            }
            cnt += dao.insert( this.getProperty( "insert-sql" )
                             , new GlueParameter<List<Object>>( paramList ) );
            this.logger.info( "{}건이 등록되었습니다", cnt );
```

실습#5-7

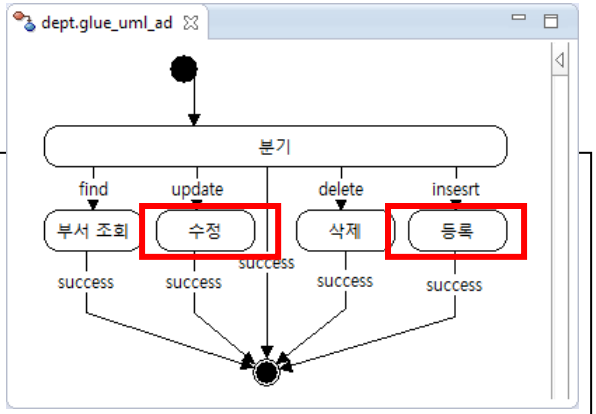
sample.activity.DeptUpsert (2/3)




```
        } else if ( "update".equals( type ) ) {
            String paramInfo
                = this.getProperty( "update-params" );
            String[] paramNames = paramInfo.split( "," );
            String loop = this.getProperty( "update-loop" );
            if ( ctx.get( loop ) != null ) {
                this.logger.debug( "수정");
                String[] checked = (String[]) ctx.get( loop );
                for ( String chk : checked ) {
                    int idx = Integer.parseInt( chk );
                    this.logger.debug( "수정 {}", idx+1);
                    List<Object> paramList = new ArrayList<Object>();
                    for ( String name : paramNames ) {
                        Object value = ctx.get( name );
                        if ( value instanceof Object[] )
                            paramList.add( ((Object[])value)[idx] );
                        else
                            paramList.add( value );
                    }
                    cnt += dao.update( this.getProperty( "update-sql" )
                                     , new GlueParameter<List<Object>>( paramList ) );
                }
                this.logger.info( "{}건이 수정되었습니다", cnt );
```

실습#5-7

sample.activity.DeptUpsert (3/3)




```
            }else {
                List<Object> paramList
                    = new ArrayList<Object>();
                for ( String name : paramNames ) {
                    Object value = ctx.get( name );
                    if ( value instanceof Object[] )
                        paramList.add( ((Object[])value)[0] );
                    else
                        paramList.add( value );
                }
                cnt += dao.update( this.getProperty( "update-sql" )
                                   , new GlueParameter<List<Object>>( paramList ) );
                this.logger.info( "{}건이 수정되었습니다", cnt );
            }
            ctx.put( this.getProperty( "result-key" ), cnt );
        }else {
            this.logger.warn( "{}은 허용되지 않습니다.", type );
            ctx.put( this.getProperty( "result-key" ), "not supported" );
        }
        ctx.put( this.getProperty( "result-key" ), cnt );
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#5-8

테스트용

```
import com.poscoict.glueframework.biz.control.GlueBizControlConstants;
import com.poscoict.glueframework.biz.control.GlueBizController;
import com.poscoict.glueframework.biz.control.GlueBizProvider;
import com.poscoict.glueframework.context.GlueContext;
import com.poscoict.glueframework.context.GlueDefaultContext;

public class Test_hello
{
    public static void main( String[] args )
    {
        GlueContext ctx = new GlueDefaultContext();
        ctx.put( GlueBizControlConstants.SERVICE_NAME, "hello-service" );
        ctx.put( "input", "Glue" );

        GlueBizController controller = GlueBizProvider.getController();
        controller.doAction( ctx );

        System.out.println( ctx.get( "HelloActivity_result" ) );
    }
}
```

- 개발에 필요한 설계 산출물
  - 화면 (Html/JSP) 정의서
  - I/O 정의서
  - Activity Diagram
  - xx-query.glue_sql
- 사전지식
  - Servlet
  - Struts framework / Spring Web MVC
  - Portlet
  - Spring Portlet MVC
  - html, java script, jsp

I/O 정의서 상세 항목 설명

- Service Name
- 화면 항목명
- 출력 사항 정의

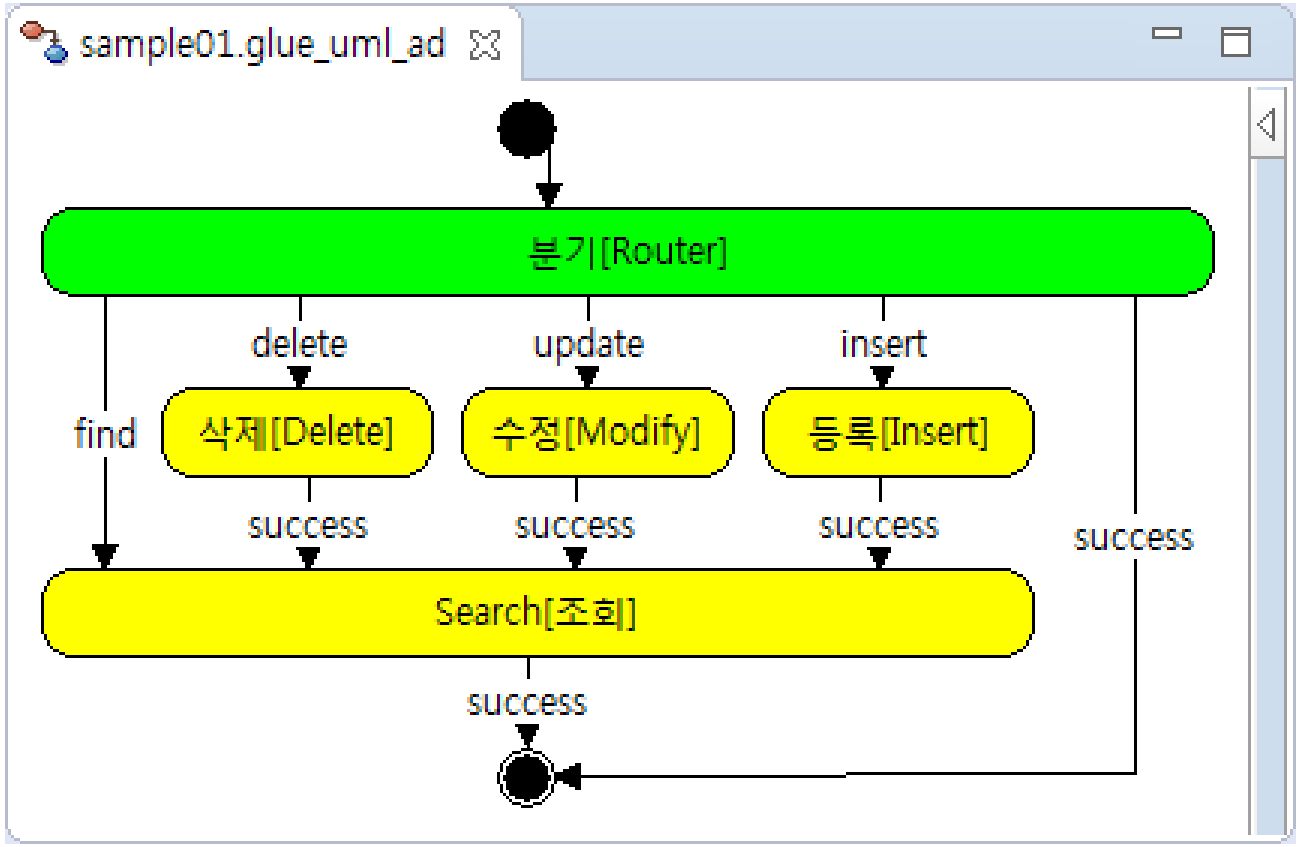
화면 입출력사양서

Chain 명 (ID)	사원 관리(M00)		Process명			작성자	이영래	수정자
Task 명			Task유형명			작성일자	06.10.21	수정일자

화 면 명	사원 관리
Service Name	Emp-service
개 요	사원의 조회와 수정
화면요구사양	하나의 화면에서 모든 작업

구 분	No.	화 면 내 용					Display 항목 정의			
		화면 항목명		입출력 구분	표현방법	Result Key	출력 사양 정	관련Table		
								Table 명	표준항목명	Type (Length)
공통	1	조회	find		IMG Button		find.x Transition Event			
	2	등록	enter		IMG Button		enter.x Transition Event			
	3	수정	modify		IMG Button		modifiy.x Transition Event			
	4	삭제	delete		IMG Button		delete.x Transition Event			
	5	부서	DeptnoP	I	Text		emp.select의 deptno=:1 입력값			
	6	EMPNO	EmpnoInser	I	Text		emp.insert의 입력 Param			
	7	ENAME	EnameInser	I	Text		emp.insert의 입력 Param			
	8	JOB	JobInsert	I	Text		emp.insert의 입력 Param			

- Activity Diagram
  - Service 로 구현할 대상.



- 주요 컴포넌트
  - **Web Controller**
    - Spring Web MVC Framework의 DispatcherServlet
  - **Biz Controller**
  - **Service**
  - **Activity**
  - **DAO**
  - **Transaction Manager**
  - **Query Manager**
  - **Cache Manager**

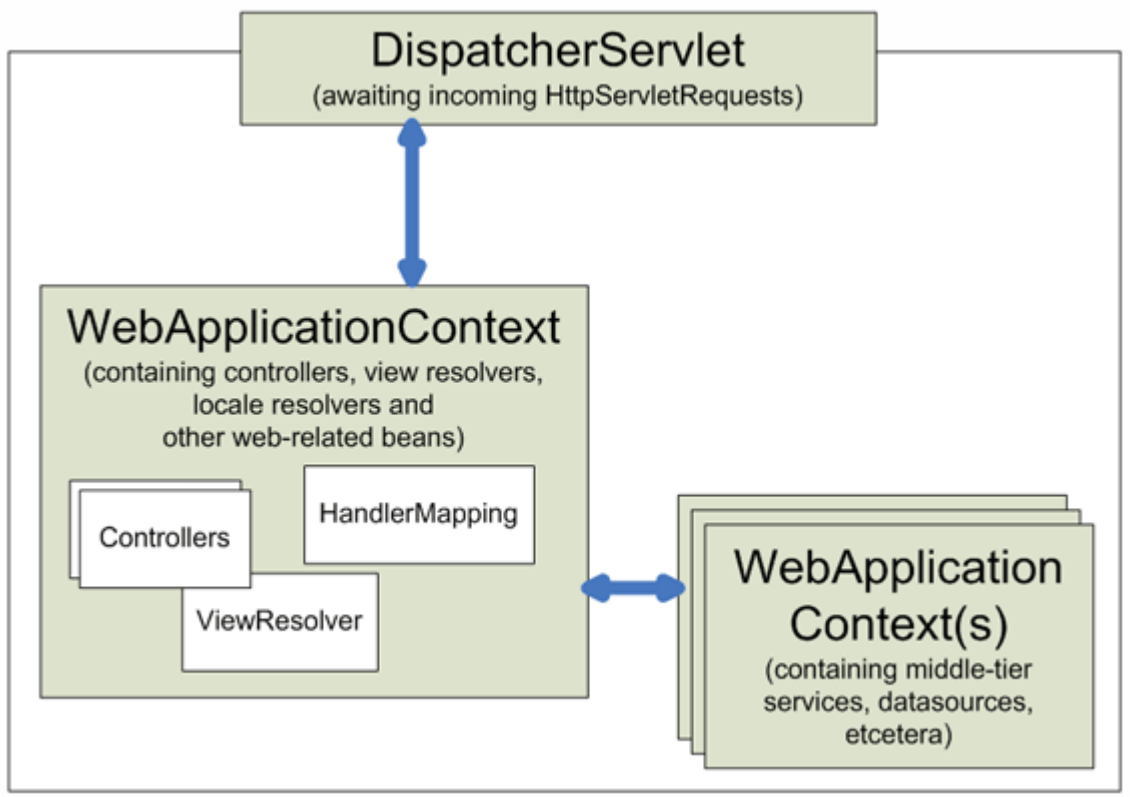


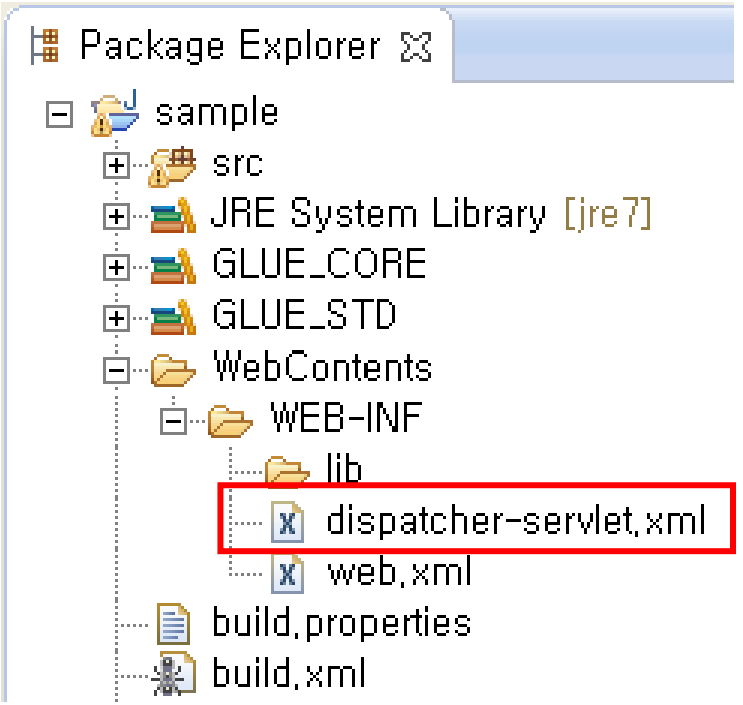
그림 출처 : http://static.springsource.org/spring/docs/3.2.x/spring-framework-reference/html/mvc.html

- web.xml 구성 (spring mvc)
  - spring Web MVC Framework 사용시 Dispatcher Servlet 추가.
    - org.springframework.web.servlet.DispatcherServlet

```
<servlet>
    <servlet-name>dispatcher</servlet-name>
    <servlet-class>
        org.springframework.web.servlet.DispatcherServlet
    </servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>dispatcher</servlet-name>
    <url-pattern>*.sp</url-pattern>
</servlet-mapping>
```

- **web.xml 구성 (spring mvc)**
  - **Dispatcher Servlet 의 configuration 파일 추가.**
    - 파일명 : <servlet-name>-servlet.xml
    - Controller, HandlerMapping, ViewResolver 를 포함함.

```
<bean name="controller"
    class="com.poscoict.glueframework.web.control.spring.GlueSimpleController" />
<bean id="beanNameUrlMapping"
    class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
    <property name="mappings">
        <value>
            /*.sp=controller
        </value>
    </property>
</bean>
<bean id="viewResolver"
    class="org.springframework.web.servlet.view.In
    <property name="prefix" value="/" />
    <property name="suffix" value=".jsp" />
</bean>
```

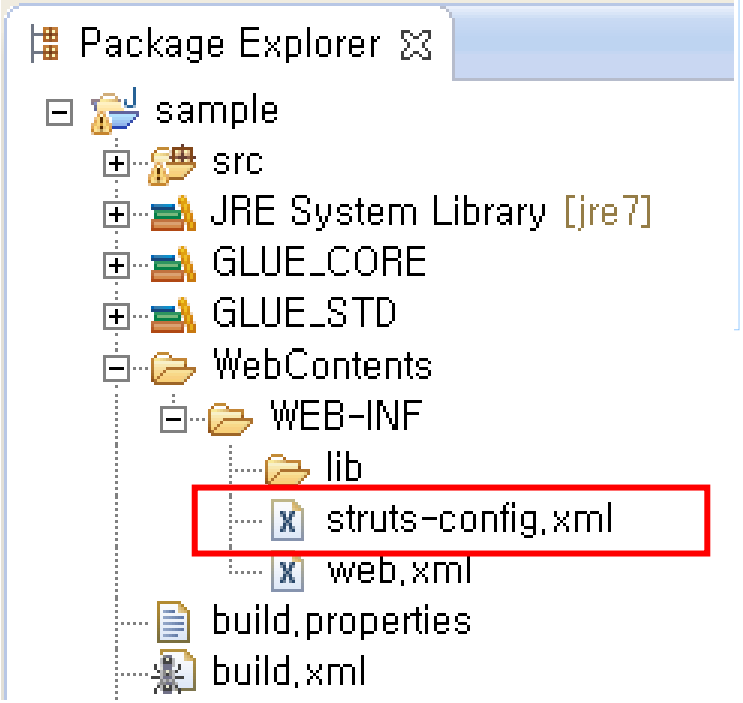


- web.xml 구성 (struts)
  - struts Framework 사용시 Action Servlet 추가.
    - org.apache.struts.action.ActionServlet

```
<servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>
        org.apache.struts.action.ActionServlet
    </servlet-class>
    <init-param>
        <param-name>config</param-name>
        <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

- web.xml 구성 (struts)
  - Action Servlet 의 config 파일 추가.
    - 파일명 : config init-param의 값

```
<action path="/sample1"
 type="com.poscoict.glueframework.web.control.struts.GlueSimpleAction">
    <forward name="success" path="/sample1.jsp"/>
    <forward name="export" path="/sample1_export.jsp"/>
    <forward name="import" path="/sample1_import.jsp"/>
</action>
<action path="/sample2"
 type="com.poscoict.glueframework.web.contro
    <forward name="success" path="/sample2.
</action>
```



- 실습#6
  - 라이브러리 추가

```
<dependency>
    <groupId>org.codehaus.jackson</groupId>
    <artifactId>jackson-mapper-asl</artifactId>
    <version>1.9.12</version>
</dependency>
```

  - web.xml
  - dispatcher-json-servlet.xml
  - dept.jsp (실습#6-3)
  - dept_jquery.jsp 추가 (#실습6-4)

실습#6-1

web.xml

```
<web-app . . .>

    <servlet>
        <servlet-name>dispatcher-json</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>dispatcher-json</servlet-name>
        <url-pattern>*.json</url-pattern>
    </servlet-mapping>

</web-app>
```

실습#6-2

dispatcher-json-servlet.xml (1/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:dwr="http://www.directwebremoting.org/schema/spring-dwr"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd
        http://www.directwebremoting.org/schema/spring-dwr
        http://www.directwebremoting.org/schema/spring-dwr-3.0.xsd">

    <bean id="urlMapping"
          class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
        <property name="mappings">
            <value>
                /*.json=controller
            </value>
        </property>
    </bean>
```

실습#6-2

dispatcher-json-servlet.xml (2/2)

```
    <bean id="controller"
          class="com.poscoict.glueframework.web.control.spring.GlueJsonController">
    </bean>
    <bean id="viewResolver"
          class="org.springframework.web.servlet.view.BeanNameViewResolver"/>
    <bean name="jsonView"
          class="org.springframework.web.servlet.view.json.MappingJacksonJsonView"/>
</beans>
```

실습#6-3

dept.jsp (1/2)

```
<%@ page contentType="text/html;charset=EUC-KR"%>
<%@ page import="com.poscoict.glueframework.context.GlueContext" %>
<%@ page import="com.poscoict.glueframework.web.control.GlueWebConstants" %>
<%@ page import="java.util.*" %>
<html>
<head>
<title>실습</title>
<meta http-equiv="Content-Type" content="text/html;charset=EUC-KR">
<body bgcolor="#FFFFFF" text="#000000" topmargin=10 leftmargin=10>
<form name="form1" method="post" action="dept.mvc">
  <input type="hidden" name="ServiceName" value="dept-service">
  <input type=submit name="find" value="조회"><hr>
</form>
<form name="form2" method="post" action="dept.mvc">
  <input type="hidden" name="ServiceName" value="dept-service">
  <input type=text name='DEPTNO' value='' size=2>
  | <input type=text name='DNAME' value='' size=14>
  | <input type=text name='LOC' value='' size=13>
  <input type=submit name="insert" value="등록"><hr>
</form><%
    GlueContext ctx = (GlueContext) request.getAttribute(GlueWebConstants.CONTEXT);
    Throwable t = ctx!=null ? ctx.getException() : null;
%><%= t==null ? "" : "[ERROR]" + t.getMessage() %><%
    List<Map> rowSet = ctx!=null ? (List<Map>)ctx.get("DeptList") : null;
%>
```

실습#6-3

dept.jsp (2/2)

```
<form name="form3" method="post" action="dept.mvc">
  <input type="hidden" name="ServiceName" value="dept-service"><%
    if(rowSet!=null){
        int i=0;
        for ( Map row : rowSet ) {
%>
  <input type=text name='DEPTNO' value='<%=row.get("DEPTNO") %>' size=2 readonly>
| <input type=text name='DNAME' value='<%=row.get("DNAME")  %>' size=14>
| <input type=text name='LOC' value='<%=row.get("LOC")    %>' size=13>
| <input type=checkbox name='chk' value='<%=i++%>'><%
            if(i==rowSet.size()){ %>
  <input type=submit name="update" value="수정"><hr><%
            }else{ %><br><%}
        }
    }
%>
</form>
<form name="form4" method="post" action="dept.mvc">
  <input type="hidden" name="ServiceName" value="dept-service">
  <input type=text name='DEPTNO' value='' size=2>
  <input type=submit name="delete" value="삭제">
</form>
</body>
</html>
```

실습#6-4

dept_jquery.jsp (1/7)

```
<!DOCTYPE html>
<%@ page contentType="text/html;charset=EUC-KR"%>
<%@ page import="com.poscoict.glueframework.context.GlueContext" %>
<%@ page import="com.poscoict.glueframework.web.control.GlueWebConstants" %>
<%@ page import="java.util.*" %>
<html>
<head>
<title>dept(jquery 사용)</title>
<meta name="viewport" content="width=device-width">
<link rel="stylesheet"
      href="http://code.jquery.com/ui/1.10.4/themes/smoothness/jquery-ui.css">
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="http://code.jquery.com/ui/1.10.4/jquery-ui.js"></script>
<script type="text/javascript">
<!--
function open_modify(idx){
    $( "#dialog-save input[name=DEPTNO]" ).attr( "value",
                                    $( "#data input[name=DEPTNO]" ).eq(idx).val() );
    $( "#dialog-save input[name=DNAME]" ).attr( "value",
                                    $( "#data input[name=DNAME]" ).eq(idx).val() );
    $( "#dialog-save input[name=LOC]" ).attr( "value",
                                    $( "#data input[name=LOC]" ).eq(idx).val() );
    $( "#dialog-save" ).dialog({ title: "부서정보 수정" });
    $( "#dialog-save" ).dialog( "open" );
}
```

실습#6-4

dept_jquery.jsp (2/7)

```
$(document).ready(function() {
    $( "#btn_refresh" ).button();
    $( "#btn_add" ).button();
    $( "#data" ).buttonset();
    $( "#data label" ).removeClass('ui-button-text ui-button-text-only');
    $( "#dialog-add" ).dialog({
        autoOpen: false, width: 380,
        buttons: {
            "등록": function() {
                $( "#dialog-add input[name=save]" ).attr( "value", "add" );
                $( "form#formAdd" ).submit(); }
        }
    });
    $( "#dialog-save" ).dialog({
        autoOpen: false, width: 380,
        buttons: {
            "저장": function() {
                $( "#dialog-save input[id=event]" ).attr( "name", "update" );
                $( "form#formSave" ).submit(); },
            "삭제": function() {
                $( "#dialog-save input[id=event]" ).attr( "name", "delete" );
                $( "form#formSave" ).submit(); }
        }
    });
});
```

실습#6-4

dept_jquery.jsp (3/7)

```
$(function() {
    $( "#btn_refresh" ).click(function( event ) {
        event.preventDefault();
        $( "form#formSearch" ).submit();
    });
    $( "#btn_add" ).click(function( event ) {
        event.preventDefault();
        $( "#dialog-add" ).dialog( "open" );
    });
    $( "#maintable tbody tr" ).dblclick(function () {
        var idx = $( "#maintable tbody tr" ).index(this);
        open_modify( idx );
    });
    $( "#maintable tbody tr" ).mouseover(function () {
        var idx = $( '#maintable tbody tr' ).index(this);
        $( "#maintable tbody tr" ).eq(idx).attr("style", 'color:red');
    });
    $( "#maintable tbody tr" ).mouseout(function () {
        var idx = $( "#maintable tbody tr" ).index(this);
        $( "#maintable tbody tr" ).eq(idx).attr("style", '');
    });
});
-->
</script>
```

실습#6-4

dept_jquery.jsp (4/7)

```
</head>
<body>
<div id="data">
    <form id="formSearch" method="post" action="dept_jquery.mvc">
        <input type="hidden" name="ServiceName" value="dept-service"/>
        <input type="hidden" name="find" value="*"/>
        <button id="btn_refresh">새로고침</button>
        <button id="btn_add">부서추가</button>
    </form>
    <table id="maintable">
        <thead class="ui-widget ui-widget-header">
            <tr>
                <th>DEPTNO</th>
                <th>DNAME</th>
                <th>LOC</th>
            </tr>
        </thead>
        <tbody class="ui-state-default"><%

    GlueContext ctx = (GlueContext) request.getAttribute(GlueWebConstants.CONTEXT);
    Throwable t = ctx!=null ? ctx.getException() : null;
%><%= t==null ? "" : "[ERROR]" + t.getMessage() %><%
    if(ctx!=null && ctx.get("DeptList")!=null) {
        List<Map> rowSet = (List<Map>)ctx.get("DeptList");
```

실습#6-4

dept_jquery.jsp (5/7)

```
        for ( Map row : rowSet ) {
%>
        <tr>
            <td><%=row.get("DEPTNO") %>
                <input type=hidden name=DEPTNO value="<%=row.get("DEPTNO") %>">
            </td>
            <td><%=row.get("DNAME") %>
                <input type=hidden name=DNAME value="<%=row.get("DNAME") %>">
            </td>
            <td><%=row.get("LOC") %>
                <input type=hidden name=LOC value="<%=row.get("LOC") %>">
            </td>
        </tr><%
        }
    }
%>
        </tbody>
    </table>
</div>
```

실습#6-4

dept_jquery.jsp (6/7)

```
<div id="dialog-add" title="부서정보 추가">
    <form id="formAdd" method="post" action="dept_jquery.mvc">
        <input type="hidden" name="ServiceName" value="dept-service"/>
        <input type="hidden" name="insert" value="*"/>
        <table>
            <tr>
                <td><label for="DEPTNO">DEPTNO</label></td>
                <td><input type="text" name="DEPTNO"
                    id="DEPTNO" class="ui-widget-content ui-corner-all" /></td>
            </tr>
            <tr>
                <td><label for="DNAME">DNAME</label></td>
                <td><input type="text" name="DNAME"
                    id="DNAME" class="ui-widget-content ui-corner-all" /></td>
            </tr>
            <tr>
                <td><label for="LOC">LOC</label></td>
                <td><input type="text" name="LOC"
                    id="LOC" class="ui-widget-content ui-corner-all" /></td>
            </tr>
        </table>
    </form>
</div>
```

실습#6-4

dept_jquery.jsp (7/7)

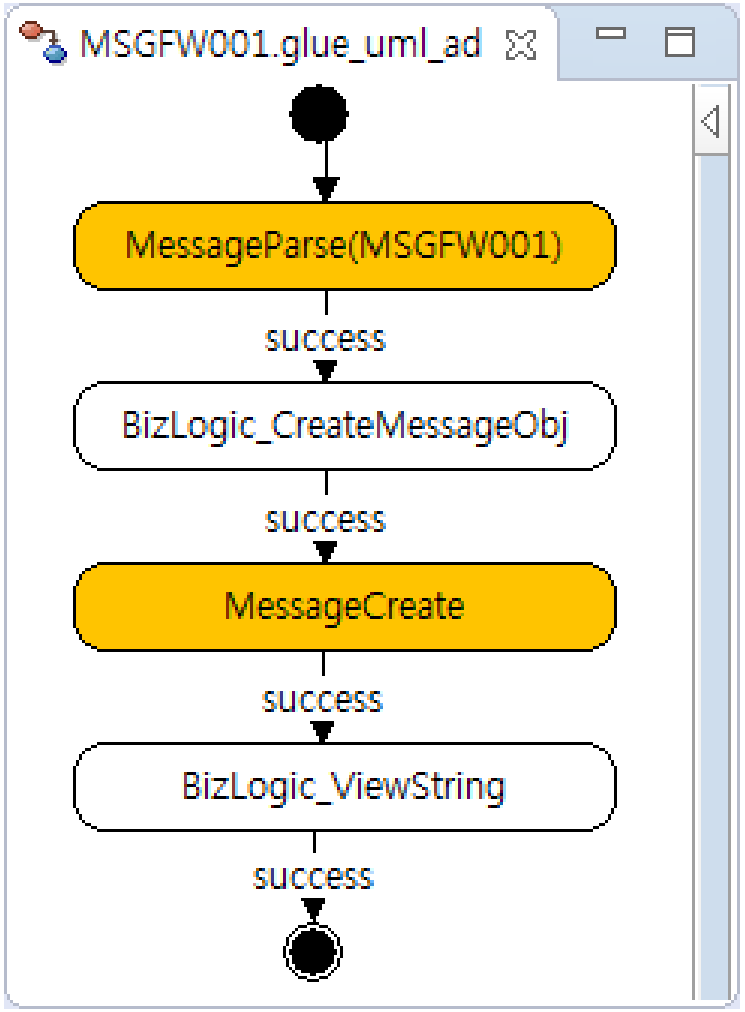
```
<div id="dialog-save" title="사용자 수정">
    <form id="formSave" method="post" action="dept_jquery.mvc">
        <input type="hidden" name="ServiceName" value="dept-service"/>
        <input type="hidden" id="event" value="any value"/>
        <table>
            <tr>
                <td><label for="DEPTNO">DEPTNO</label></td>
                <td><input type="text" name="DEPTNO"
                    id="DEPTNO" class="ui-widget-content ui-corner-all" /></td>
            </tr>
            <tr>
                <td><label for="DNAME">DNAME</label></td>
                <td><input type="text" name="DNAME"
                    id="DNAME" class="ui-widget-content ui-corner-all" /></td>
            </tr>
            <tr>
                <td><label for="LOC">LOC</label></td>
                <td><input type="text" name="LOC"
                    id="LOC" class="ui-widget-content ui-corner-all" /></td>
            </tr>
        </table>
    </form>
</div>
</body>
</html>
```

- 실습
  - **http://127.0.0.1:8080/sample.mvc**
  - **http://127.0.0.1:8080/sample_vo.mvc**
  - **http://127.0.0.1:8080/sample_global.mvc**
    - glue.properties
      - default.ui.locale=ko_KR
      - default.ui.locale=en_US
  - **http://127.0.0.1:8080/dept.mvc**
  - **http://127.0.0.1:8080/dept_jquery.mvc**
  - **http://127.0.0.1:8080/dept.json?ServiceName=dept-service&find=1**

- 개발에 필요한 설계 산출물
  - Layout 정의서 : xx-msg.xml 또는 DB
  - Activity Diagram
  - xx-query.glue_sql
- Layout

no.	유형	항목명	항목ID	항목Type	길이	단위	소숫점이하
1	항목	TransactionCode	TRANSACTION_CODE	VARCHAR	8		0
2	항목	CRUD구분	CRUD	VARCHAR	1		0
3	항목	부서번호	DEPTNO	NUMBER	2		0
4	항목	부서이름	DNAME	VARCHAR	14		0
5	항목	부서위치	LOC	VARCHAR	13		0

- **Activity Diagram**
  - **Service 로 구현할 대상.**



- web.xml 구성 (http 방식)
  - HttpReceiver Servlet 추가.
    - com.poscoict.glueframework.web.GlueHttpReceiverAdapter

```
<servlet>
    <servlet-name>HttpReceiver</servlet-name>
    <servlet-class>
        com.poscoict.glueframework.web.GlueHttpReceiverAdapter
    </servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>HttpReceiver</servlet-name>
    <url-pattern>*.tc</url-pattern>
</servlet-mapping>
```

실습#7

라이브러리 추가

```
<dependency>
    <groupId>commons-httpclient</groupId>
    <artifactId>commons-httpclient</artifactId>
    <version>3.1</version>
    <scope>test</scope>
</dependency>
```

MSGFW004.dat

```
MSGFW001D90who            SEOUL          .
MSGFW001C90abcde          seoul          .
MSGFW001U90HR             PANGYO         .
```

- web.xml
- sample.activity.ValueRouter (실습#7-2)
- sample.activity.SaveDept (실습#7-3)
- sample.activity.BizLogic (실습#7-4)
- sample.activity.ChangeNext (실습#7-5)

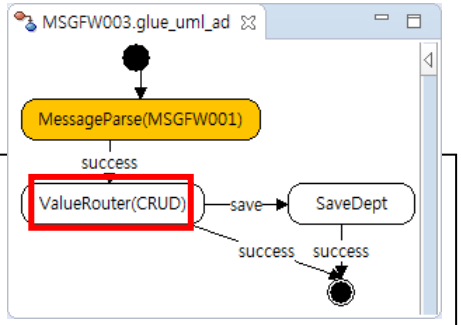
실습#7-1

web.xml

```
<web-app . . .>

    <servlet>
        <servlet-name>HttpReceiver</servlet-name>
        <servlet-class>
            com.poscoict.glueframework.web.GlueHttpReceiverAdapter
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HttpReceiver</servlet-name>
        <url-pattern>/receiver.tc</url-pattern>
    </servlet-mapping>

</web-app>
```

실습#7-2

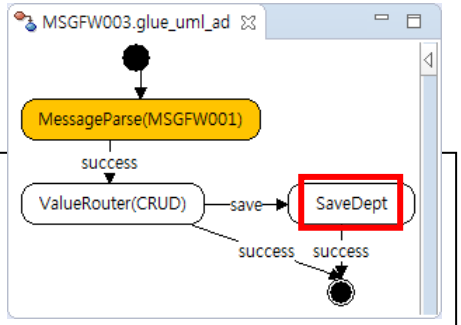
sample.activity.ValueRouter




```
package sample.activity;

import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.message.*;

public class ValueRouter extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        /* Value Router : 값에 따른 분기 */
        GlueMessage message = ctx.getMessage();
        if ( message != null ) {
            String crud = (String) message.get( "CRUD" );
            String crud2 = (String) ctx.get( "CRUD" );
            this.logger.info( "{},{}", crud, crud2 );
            if ( "C".equals( crud ) || "U".equals( crud ) || "D".equals( crud ) ){
                return "save";
            } else if ( "S".equals( crud ) ) {
                return "send"; /* ERROR */
            }
        }
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#7-3

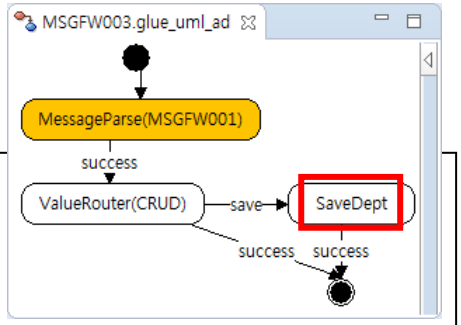
sample.activity.SaveDept (1/2)




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.dao.*;
import com.poscoict.glueframework.dao.vo.*;
import com.poscoict.glueframework.message.*;
public class SaveDept extends GlueActivity<GlueContext> {
    public String runActivity(GlueContext ctx) {
        GlueGenericDao dao = this.getDao( this.getProperty( GlueActivityConstants.DAO ) );
        GlueMessage msg = ctx.getMessage();
        String crud = (String)msg.get( "CRUD" );
        if(crud.equals( "C" )) {
            String paramInfo = this.getProperty( "insert-param" );
            String[] paramNames = paramInfo.split( "," );
            List<Object> paramList = new ArrayList<Object>();
            for ( String name : paramNames ) {
                paramList.add( msg.get( name ) );
            }
            GlueParameter<List<Object>> param
                = new GlueParameter<List<Object>>( paramList );
            int cnt = dao.insert( this.getProperty( "insert-sql" ), param );
            this.logger.info( "{}건이 추가 되었습니다.", cnt );
```

실습#7-3

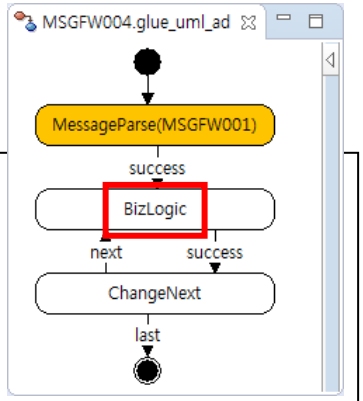
sample.activity.SaveDept (2/2)




```
        }else if(crud.equals( "U" )){
            String paramInfo = this.getProperty( "update-param" );
            String[] paramNames = paramInfo.split( "," );
            List<Object> paramList = new ArrayList<Object>();
            for ( String name : paramNames ) {
                paramList.add( msg.get( name ) );
            }
            int cnt = dao.update( this.getProperty( "update-sql" ) ,
                                  new GlueParameter<List<Object>>( paramList ));
            this.logger.info( "{}건이 수정 되었습니다.", cnt );
        }else if(crud.equals( "D" )){
            String paramInfo = this.getProperty( "delete-param" );
            String[] paramNames = paramInfo.split( "," );
            Map<String, Object> paramMap = new HashMap<String, Object> ();
            for ( String nameInfo : paramNames ) {
                String[] name = nameInfo.split( "=" );
                paramMap.put( name[0], msg.get( name[1] ) );
            }
            int cnt = dao.delete( this.getProperty( "delete-sql" ) ,
                                  new GlueParameter<Map<String, Object>>( paramMap ));
            this.logger.info( "{}건이 삭제 되었습니다.", cnt );
        }
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#7-4

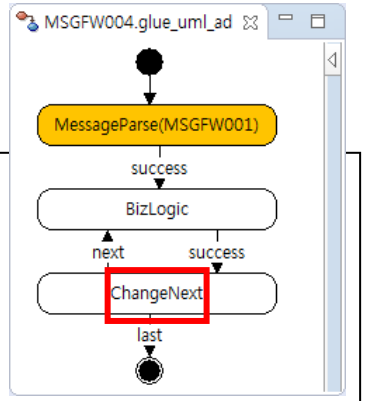
sample.activity.BizLogic




```
package sample.activity;
import java.util.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.message.*;
public class BizLogic extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        List<GlueMessage> messageList = ctx.getMessages();
        this.logger.info( "수신 : [{}]", ctx.get( GlueBizControlConstants.RECEIVE_TC ) );
        this.logger.info( "Parsing TC 개수 : {}", messageList.size() );
        int idx = ctx.get( "tc-idx" ) != null
                ? ( (Integer) ctx.get( "tc-idx" ) ).intValue()
                : 0;
        GlueMessage message = messageList.get( idx );
        this.logger.info( "TC : [{}", message.getTC() );
        /* SubService 실행 */
        String mainServiceName = (String)ctx.get(GlueBizControlConstants.SERVICE_NAME);
        ctx.putAll( ctx.getMessages().get( idx ).getAttributes() );
        ctx.put( GlueBizControlConstants.SERVICE_NAME, "dept-service" );
        GlueBizProvider.getController().doSubController( ctx, true );
        ctx.put( GlueBizControlConstants.SERVICE_NAME, mainServiceName );
        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#7-5

sample.activity.ChangeNext




```
package sample.activity;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.context.*;
public class ChangeNext extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        Object idx = ctx.get( "tc-idx" );
        if ( idx == null ){
            idx = new Integer( 0 );
        }
        int total = ctx.getMessages().size();
        int current = ( (Integer) idx ).intValue() + 1;
        ctx.put( "tc-idx", new Integer( current ) );
        if ( current == total )
        {
            return "last";
        } else
        {
            return "next";
        }
    }
}
```

실습#7-6

테스트용

```
import java.text.*;
import java.util.*;
import org.apache.commons.httpclient.*;
import org.apache.commons.httpclient.methods.*;
public class SendTC {
    public static void main(String[] args) {
        HttpClient client = new HttpClient();
        String d = new SimpleDateFormat( "yyyyMMddHHmmss" ).format( new Date());
        NameValuePair param[] = new NameValuePair[]{ new NameValuePair("ifd", "TEST_TC_TC"),
            new NameValuePair("timestamp", d),
            new NameValuePair("sequence", "1"),
            new NameValuePair("message", "MSGFW001D40.............*************"),
            new NameValuePair("type", "T")}; /* 파일은 type=F임. message는 path포함 파일명임 */
        HttpMethod method = new PostMethod("http://127.0.0.1:8080/GlueSample/receiver.tc");
        method.setQueryString(param);
        try {
            System.out.println( client.executeMethod(method) );
        } catch ( Exception e ) {
            e.printStackTrace();
        }finally{
            method.releaseConnection();
        }
    }
}
```

- SOAP services 용
  - web.xml 에 CXFServlet 추가
    - org.apache.cxf.transport.servlet.CXFServlet

```
<listener />
<context-param />
<servlet>
    <servlet-name>CXFServlet</servlet-name>
    <servlet-class>
        org.apache.cxf.transport.servlet.CXFServlet
    </servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>CXFServlet</servlet-name>
    <url-pattern>/ws/*</url-pattern>
</servlet-mapping>
```

- SOAP services 용
  - web.xml 에 ContextLoaderListener 추가.
    - org.springframework.web.context.ContextLoaderListener

```
<listener>
    <listener-class>
        org.springframework.web.context.ContextLoaderListener
    </listener-class>
</listener>
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
        /WEB-INF/jaxws-server-endpoint.xml
    </param-value>
</context-param>
<servlet />
```

- SOAP services 용
  - Configuraion 파일 (jaxws-server-endpoint.xml) 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans . . .>

 <import resource="classpath:META-INF/cxf/cxf.xml"/>
 <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml"/>
 <import resource="classpath:META-INF/cxf/cxf-servlet.xml"/>

 <bean id="glueJaxWebServiceImpl"
   class="com.poscoict.glueframework.jaxws.GlueJaxWebServiceImpl">
 </bean>
 <jaxws:endpoint id="glueJaxWebService"
       implementor="#glueJaxWebServiceImpl" address="/glueJaxWs"/>
</beans>
```

  - endpoint
    - http://localhost/context/ws/glueJaxWs

- Configuraion 파일 (client)
  - applicationContext.xml 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans . . .>

 <jaxws:client id="glueJaxWsService"
  serviceClass="com.poscoict.glueframework.jaxws.GlueJaxWebService"
  address="http://localhost/context/ws/glueJaxWs" />

</beans>
```

- 실습#8
  - Library 추가

```
<dependency>
    <groupId>com.poscoict</groupId>
    <artifactId>glue-ws</artifactId>
    <version>${glue.version}</version>
</dependency>
<dependency>
    <groupId>org.apache.cxf</groupId>
    <artifactId>cxf-rt-frontend-jaxws</artifactId>
    <version>2.7.14</version>
</dependency>
<dependency>
    <groupId>org.apache.cxf</groupId>
    <artifactId>cxf-rt-transports-http</artifactId>
    <version>2.7.14</version>
</dependency>
<dependency>
    <groupId>com.sun.xml.bind</groupId>
    <artifactId>jaxb-impl</artifactId>
    <version>2.2.6</version>
</dependency>
```

  - web.xml
  - jaxws-server-endpoint.xml
  - sample.activity.SearchEmp (실습#8-3)

실습#8-1

web.xml

```
<web-app . . .>

    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/jaxws-server-endpoint.xml</param-value>
    </context-param>
    <servlet>
        <servlet-name>CXFServlet</servlet-name>
        <servlet-class>org.apache.cxf.transport.servlet.CXFServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>CXFServlet</servlet-name>
        <url-pattern>/ws/*</url-pattern>
    </servlet-mapping>

</web-app>
```

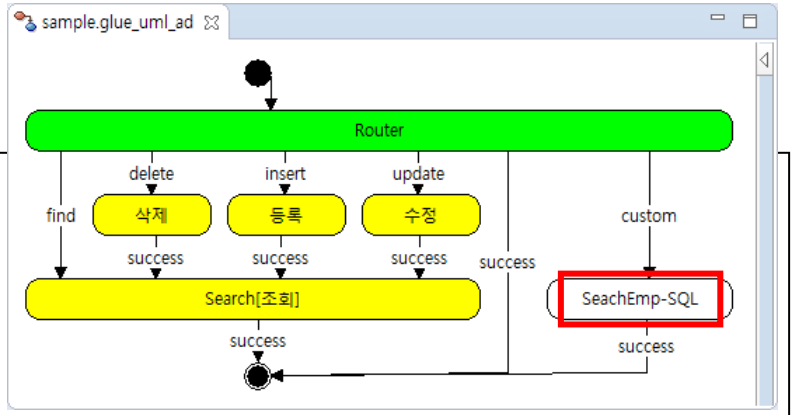
실습#8-2

jaxws-server-endpoint.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:cxf="http://cxf.apache.org/core" xmlns:jaxws="http://cxf.apache.org/jaxws"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd
        http://cxf.apache.org/core
        http://cxf.apache.org/schemas/core.xsd
        http://cxf.apache.org/jaxws
        http://cxf.apache.org/schemas/jaxws.xsd">

    <import resource="classpath:META-INF/cxf/cxf.xml"/>
    <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml"/>
    <import resource="classpath:META-INF/cxf/cxf-servlet.xml"/>

    <bean id="glueJaxWebServiceImpl"
          class="com.poscoict.glueframework.jaxws.GlueJaxWebServiceImpl"/>
    <jaxws:endpoint id="glueJaxWebService"
          implementor="#glueJaxWebServiceImpl"
          address="/glueJaxWs" />
</beans>
```

실습#8-3

sample.activity.SearchEmp (1/2)




```
package sample.activity;
import java.util.*;
import sample.vo.*;
import com.poscoict.glueframework.biz.activity.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.dao.*;
import com.poscoict.glueframework.dao.vo.*;
public class SearchEmp extends GlueActivity<GlueContext> {
    public String runActivity( GlueContext ctx ) {
        GlueGenericDao dao = this.getDao( "test-dao" );

        List<String> list = new ArrayList<String>();
        list.add( "10" );
        GlueParameter<List<String>> listParam = new GlueParameter<List<String>>();
        listParam.setParameter( list );

        Map<String, Object> map = new HashMap<String, Object>();
        map.put( "deptno", "10" );
        GlueParameter<Map<String, Object>> mapParam
            = new GlueParameter<Map<String, Object>>();
        mapParam.setParameter( map );
```

실습#8-3

sample.activity.SearchEmp (2/2)

```
        List<Map<String, Object>> result1 = dao.find( "sample.emp.select", listParam );
        List<EmpVO> result2 = dao.find( "sample.emp.select.vo", listParam );
        List<Map<String, Object>> result3 = dao.find( "sample.emp.select.named", mapParam );
        List<EmpVO> result4 = dao.find( "sample.emp.select.named.vo", mapParam );

        ctx.put( "EmpList", result1 );
        ctx.put( "EmpListUsingVO", result2 );
        ctx.put( "EmpList_named", result3 );
        ctx.put( "EmpList_namedUsingVO", result4 );

        ArrayList<String> empno = new ArrayList<String>();
        for ( Map<String, Object> row : result1 ) {
            empno.add( ""+row.get( "EMPNO" ) );
        }
        ArrayList<String> ename = new ArrayList<String>();
        for ( Map<String, Object> row : result3 ) {
            ename.add( ""+row.get( "ENAME" ) );
        }
        ctx.put( "ws-empnoList", empno);
        ctx.put( "ws-enameList", ename);
        ctx.put( "ws-EmpList1", new HashMap<String, Object>( result1.get( 0 ) ) );

        return GlueBizControlConstants.SUCCESS;
    }
}
```

실습#8-4

applicationContext.xml (sample-maven/src/test/resource/)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:jaxws="http://cxf.apache.org/jaxws"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://cxf.apache.org/jaxws
        http://cxf.apache.org/schemas/jaxws.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd">

    <jaxws:client id="ws"
        serviceClass="com.poscoict.glueframework.jaxws.GlueJaxWebService"
        address="http://127.0.0.1:8080/GlueSample/ws/glueJaxWs" />
</beans>
```

실습#8-5

CallWebService.java (1/2)

```
import java.util.*;
import com.poscoict.glueframework.biz.control.*;
import com.poscoict.glueframework.context.*;
import com.poscoict.glueframework.jaxws.*;
public class CallWebService {
    public static void main( String[] args ) {
        try {
            GlueJaxWebService ws = GlueStaticContext.getBeanFactory(
                ).getBeanObject( "ws", GlueJaxWebService.class );

            HashMap<String, Object> glueMap = new HashMap<String, Object>();
            glueMap.put( "ServiceName", "sample-service" );
            glueMap.put( "custom", "1" );
            glueMap.put( GlueBizControlConstants.JAXWS_RESULT, "ServiceName" );
            String rString = ws.doGlueServiceGetString( glueMap );
            System.out.println("String : " + rString);

            glueMap = new HashMap<String, Object>();
            glueMap.put( "ServiceName", "sample-service" );
            glueMap.put( "custom", "1" );
            glueMap.put( GlueBizControlConstants.JAXWS_RESULT, "ws-empnoList" );
            List<String> rList = (List<String>) ws.doGlueServiceGetList( glueMap );
            System.out.println("List : " + rList);
```

실습#8-5

CallWebService.java (2/2)

```
            glueMap = new HashMap<String, Object>();
            glueMap.put( "ServiceName", "sample-service" );
            glueMap.put( "custom", "1" );
            glueMap.put( GlueBizControlConstants.JAXWS_RESULT, "ws-EmpList1" );
            HashMap<String, Object> rMap
                = (HashMap<String, Object>) ws.doGlueServiceGetMap( glueMap );
            System.out.println("Map : " + rMap);
        } catch ( Exception e )
        {
            e.printStackTrace();
        }
    }
}
```

- Spring Scheduler 사용시
  - spring_scheduler.xml 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans . . .">
  <task:scheduled-tasks scheduler="scheduler">
    <task:scheduled ref="gluetask" method="doGlueService"
            fixed-delay="5000"/>
    <!-- <task:scheduled ref="gluetask" method="doGlueService"
            cron="*/8 * * * * MON-FRI"/> -->
  </task:scheduled-tasks>
  <task:scheduler id="scheduler" pool-size="3"/>
  <bean id="gluetask"
        class="c~.p~.g~.scheduling.task.GlueTaskScheduler">
    <property name="contextMap">
      <map>
        <entry key="ServiceName" value="sample-service"/>
        <entry key="deptno" value="10"/>
      </map>
    </property>
  </bean>
</beans>
```

- Quartz 사용시
  - quartz_scheduler.xml 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
 "http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="scheduler"
 class="org.springframework.scheduling.quartz.SchedulerFactoryBean"
 lazy-init="true">
    <property name="triggers">
      <list>
        <ref local="simpleTrigger"/>
        <ref local="cronTrigger"/>
      </list>
    </property>
  </bean>
```

- Quartz 사용시
  - quartz_scheduler.xml 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="scheduler" . . ./>
  <bean id="simpleTrigger"
   class="org.springframework.scheduling.quartz.SimpleTriggerBean">
    <property name="jobDetail" ref="jobDetail"/>
    <property name="repeatInterval" value="10000"/>
    <property name="repeatCount" value="3"/>
  </bean>
  <bean id="cronTrigger"
   class="org.springframework.scheduling.quartz.CronTriggerBean">
    <property name="jobDetail" ref="jobDetail"/>
    <!-- cronExpression : 초 분 시 날 달 주 -->
    <property name="cronExpression" value="0 14 15 * * ?"/>
  </bean>
```

- Quartz 사용시
  - quartz_scheduler.xml 추가.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="scheduler" . . ./>
  <bean id="simpleTrigger" . . ./>
  <bean id="cronTrigger" . . ./>
  <bean id="jobDetail"
    class="org.springframework.scheduling.quartz.JobDetailBean">
    <property name="jobClass"
              value="c~.p~.g~.scheduling.GlueQuartzJobBean"/>
    <property name="jobDataAsMap">
      <map>
        <entry key="ServiceName" value="sample-service"/>
        <entry key="deptno" value="10"/>
      </map>
    </property>
  </bean>
</beans>
```

Cron 표현식

7개의 단위 표현식으로 구성된 문자열

- 초 (Seconds) – 0에서 59 사이의 숫자
- 분 (Minutes) – 0에서 59 사이의 숫자
- 시 (Hours) – 0에서 23 사이의 숫자
- 일 (Day of month) – 1에서 31 사이의 숫자
- 월 (Months) – 1에서 12사이의 숫자, 또는 JAN, FEB, .. , DEC
- 주 (Days of week) – 1에서 7사이의 숫자, 또는 MON, TUE, .. , SUN
- 연 (Year) - 생략 가능
- 특수문자 : [*][?][-][,][/][L]

```
0 15 12 * * ? : 매일 12시 15분마다 실행 (모든값[*], 설정값없음[?])
0 15 12 ? * * 2015 : 2015년도에 모든 요일마다 12시 15분마다 실행
0 15 12 * * ? 2010-2014 : 2010~14년도에 매일 12시 15분마다 실행(범위[-])
0 0 8 ? * MON,FRI : 월금 8시마다(나열[,])
0 0/30 8-9 * * * : 매일 8시 10시 사이에 30분 간격으로 실행(간격[/])
                   8:00, 8:30, 9:00, 9:30
0 0 23 L * ? : 매월 마지막날 23시에 실행 [마지막일, 마지막요일[L])
```

- 실습#9
  - pom.xml
  - 라이브러리 추가

```
<dependency>
    <groupId>com.poscoict</groupId>
    <artifactId>glue-schedule</artifactId>
    <version>${glue.version}</version>
</dependency>
<dependency>
    <groupId>org.quartz-scheduler</groupId>
    <artifactId>quartz</artifactId>
    <version>1.8.6</version>
</dependency>
<dependency>
    <groupId>org.apache.mina</groupId>
    <artifactId>mina-core</artifactId>
    <version>1.0.10</version>
</dependency>
```

    - spring-context 순서 조정
  - spring-scheduler.xml 추가
  - quartz-scheduler.xml 추가

실습#9-1

pom.xml (1/2)

```
<project . . .
    <packaging>jar</packaging>
. . .
    <build>
        <finalName>user-application</finalName>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-jar-plugin</artifactId>
                <version>2.5</version>
                <configuration>
                    <archive>
                        <manifest>
                            <addClasspath>true</addClasspath>
                            <classpathPrefix>lib/</classpathPrefix>
                        </manifest>
                        <manifestEntries>
                            <Main-Class>
                      com.poscoict.glueframework.scheduling.server.GlueSchedulerHttpServer
                            </Main-Class>
                        </manifestEntries>
                    </archive>
                </configuration>
            </plugin>
```

실습#9-1

pom.xml (1/2)

```
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-dependency-plugin</artifactId>
                <version>2.9</version>
                <executions>
                    <execution>
                        <id>copy-dependencies</id>
                        <phase>package</phase>
                        <goals>
                            <goal>copy-dependencies</goal>
                        </goals>
                        <configuration>
                            <outputDirectory>
                                ${project.build.directory}/lib
                            </outputDirectory>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
            . . . maven-compiler-plugin
        </plugins>
    </build>
    . . . properties & dependencies
</project>
```

실습#9-2

spring-scheduler.xml (1/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:task="http://www.springframework.org/schema/task"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
                        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
                        http://www.springframework.org/schema/task
                        http://www.springframework.org/schema/task/spring-task.xsd">

    <task:scheduled-tasks scheduler="scheduler">
        <task:scheduled ref="glue-task-1" method="doGlueService"
                                          cron="0/8 * * * * MON-FRI"/>
        <task:scheduled ref="glue-task-2" method="doGlueService" fixed-rate="70000"/>
        <task:scheduled ref="glue-task-3" method="doGlueService" fixed-delay="50000"/>
    </task:scheduled-tasks>

    <task:scheduler id="scheduler" pool-size="3" />

    <bean id="glue-task-1"
          class="com.poscoict.glueframework.scheduling.task.GlueTaskScheduler">
        <property name="ServiceName" value="hello-service"/>
    </bean>
```

실습#9-2

spring-scheduler.xml (2/2)

```
    <bean id="glue-task-2"
          class="com.poscoict.glueframework.scheduling.task.GlueTaskScheduler">
        <property name="ServiceName" value="dept-service"/>
        <property name="dataMap">
            <map>
                <entry key="DEPTNO" value="50"/>
                <entry key="DNAME" value="Marketing"/>
                <entry key="LOC" value="SEOUL"/>
                <entry key="insert" value="event"/>
            </map>
        </property>
    </bean>
    <bean id="glue-task-3"
          class="com.poscoict.glueframework.scheduling.task.GlueTaskScheduler">
        <property name="ServiceName" value="dept-service"/>
        <property name="dataMap">
            <map>
                <entry key="DEPTNO" value="50"/>
                <entry key="delete" value="event"/>
            </map>
        </property>
    </bean>
</beans>
```

실습#9-3

quartz-scheduler.xml (1/3)

```
<?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/spring-beans-3.0.xsd">
    <bean id="scheduler"
          class="org.springframework.scheduling.quartz.SchedulerFactoryBean" >
        <property name="triggers">
            <list>
                <ref local="simpleTrigger1"/>
                <ref local="simpleTrigger2"/>
                <ref local="cronTrigger"/>
            </list>
        </property>
    </bean>

    <bean id="cronTrigger" class="org.springframework.scheduling.quartz.CronTriggerBean">
        <property name="jobDetail" ref="jobDetail-A"/>
        <property name="cronExpression" value="0 40 15 * * ?"/>
    </bean>
```

실습#9-3

quartz-scheduler.xml (2/3)

```
<bean id="simpleTrigger1"
      class="org.springframework.scheduling.quartz.SimpleTriggerBean">
    <property name="jobDetail" ref="jobDetail-B"/>
    <property name="repeatInterval" value="10000"/>
    <property name="repeatCount" value="3"/>
    <property name="startDelay" value="5000"/>
</bean>
<bean id="simpleTrigger2"
      class="org.springframework.scheduling.quartz.SimpleTriggerBean">
    <property name="jobDetail" ref="jobDetail-C"/>
    <property name="repeatInterval" value="7000"/>
</bean>

<bean id="jobDetail-A" class="org.springframework.scheduling.quartz.JobDetailBean">
    <property name="jobClass"
              value="com.poscoict.glueframework.scheduling.GlueQuartzJobBean"/>
    <property name="jobDataAsMap">
        <map>
            <entry key="ServiceName" value="hello-service"/>
        </map>
    </property>
</bean>
```

실습#9-3

quartz-scheduler.xml (3/3)

```
    <bean id="jobDetail-B" class="org.springframework.scheduling.quartz.JobDetailBean">
        <property name="jobClass"
                  value="com.poscoict.glueframework.scheduling.GlueQuartzJobBean"/>
        <property name="jobDataAsMap">
            <map>
                <entry key="ServiceName" value="dept-service"/>
                <entry key="DEPTNO" value="60"/>
                <entry key="DNAME" value="Testing"/>
                <entry key="LOC" value="seoul"/>
                <entry key="insert" value="event"/>
            </map>
        </property>
    </bean>
    <bean id="jobDetail-C" class="org.springframework.scheduling.quartz.JobDetailBean">
        <property name="jobClass"
                  value="com.poscoict.glueframework.scheduling.GlueQuartzJobBean"/>
        <property name="jobDataAsMap">
            <map>
                <entry key="ServiceName" value="dept-service"/>
                <entry key="DEPTNO" value="60"/>
                <entry key="delete" value="event"/>
            </map>
        </property>
    </bean>
</beans>
```

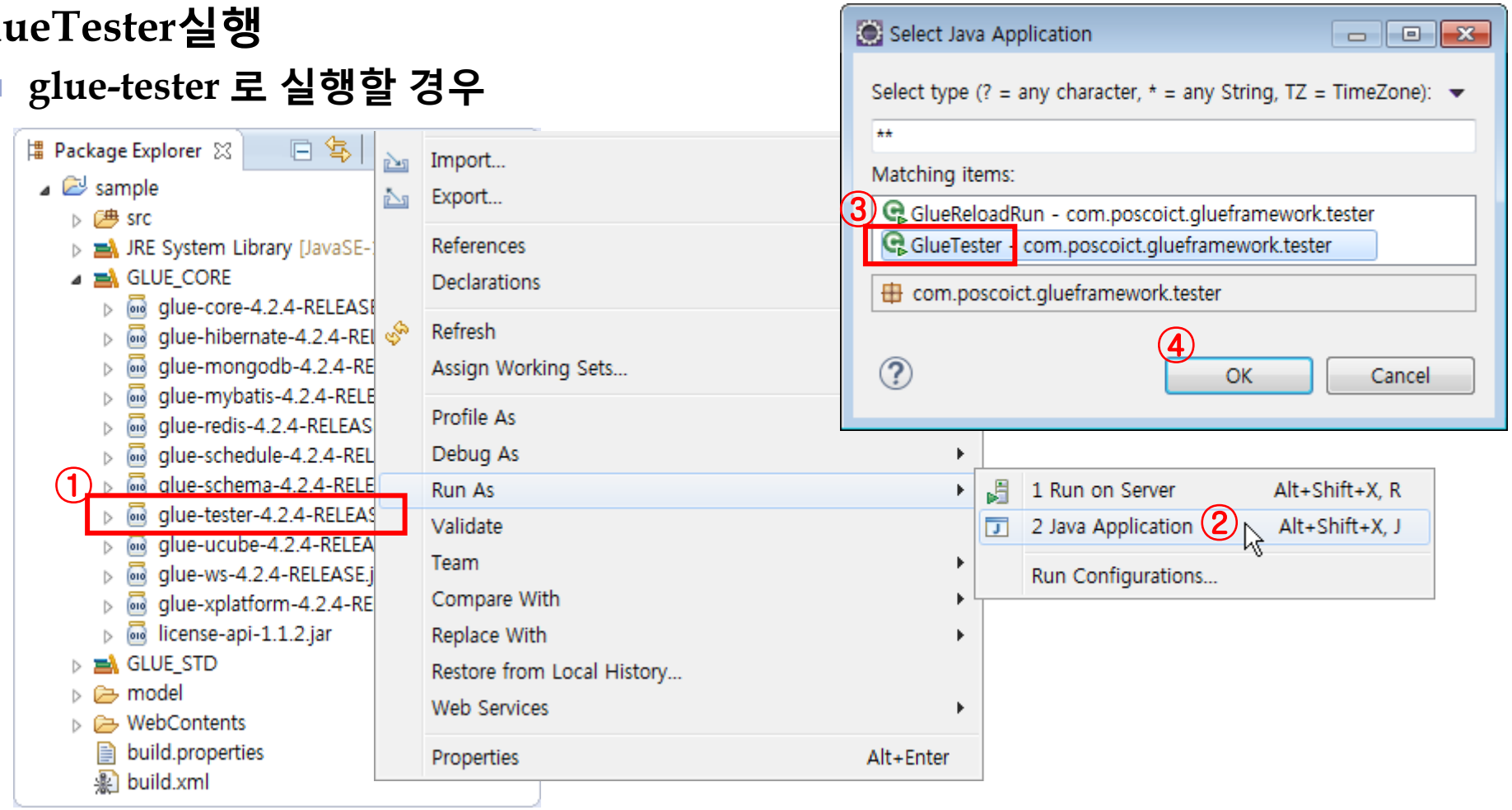
실습#9-4

실행

```
java -jar user-application.jar quartz quartz-scheduler.xml
```

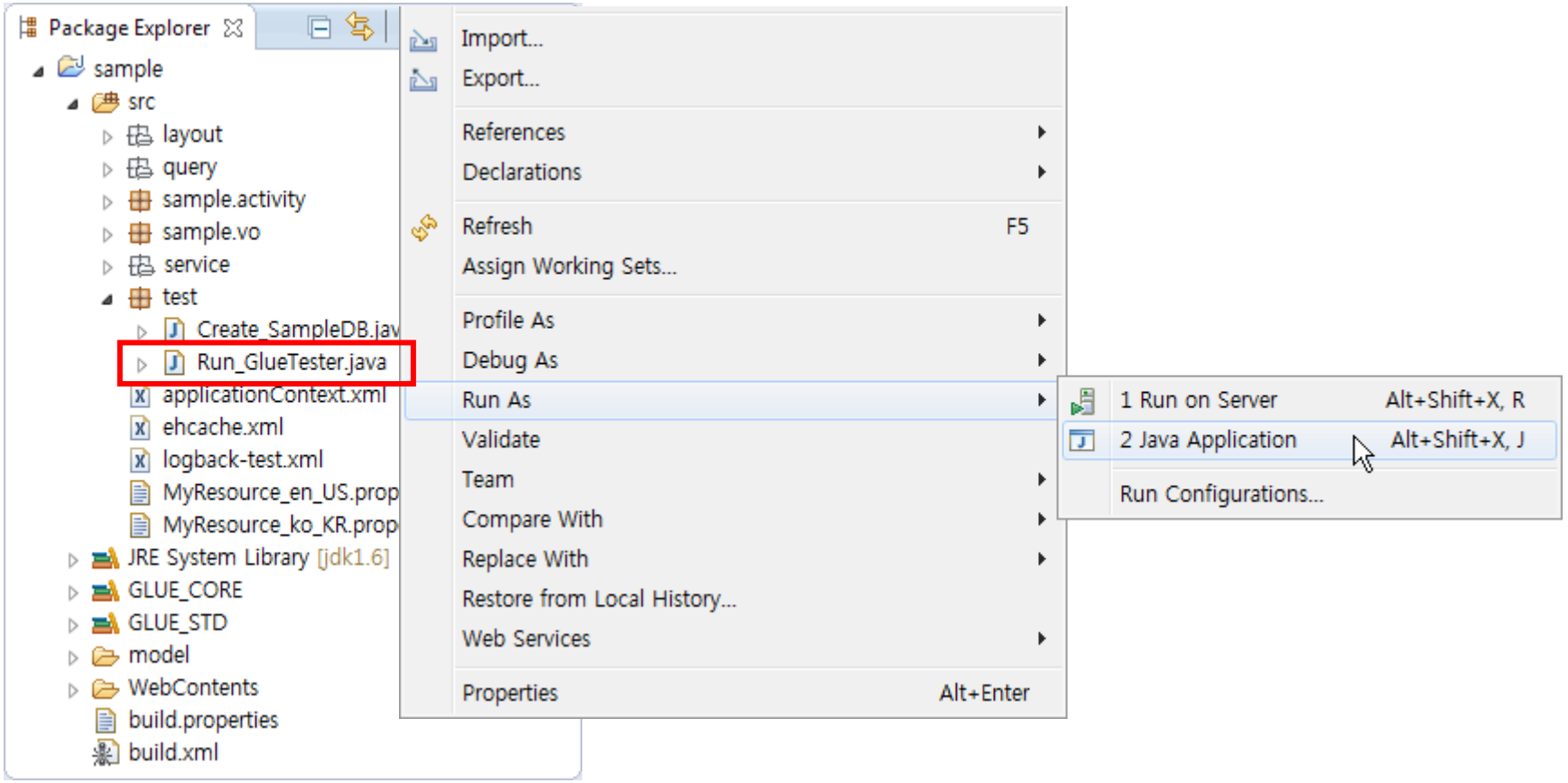
```
java -jar user-application.jar spring spring-scheduler.xml
```

- GlueTester실행
  - glue-tester 로 실행할 경우



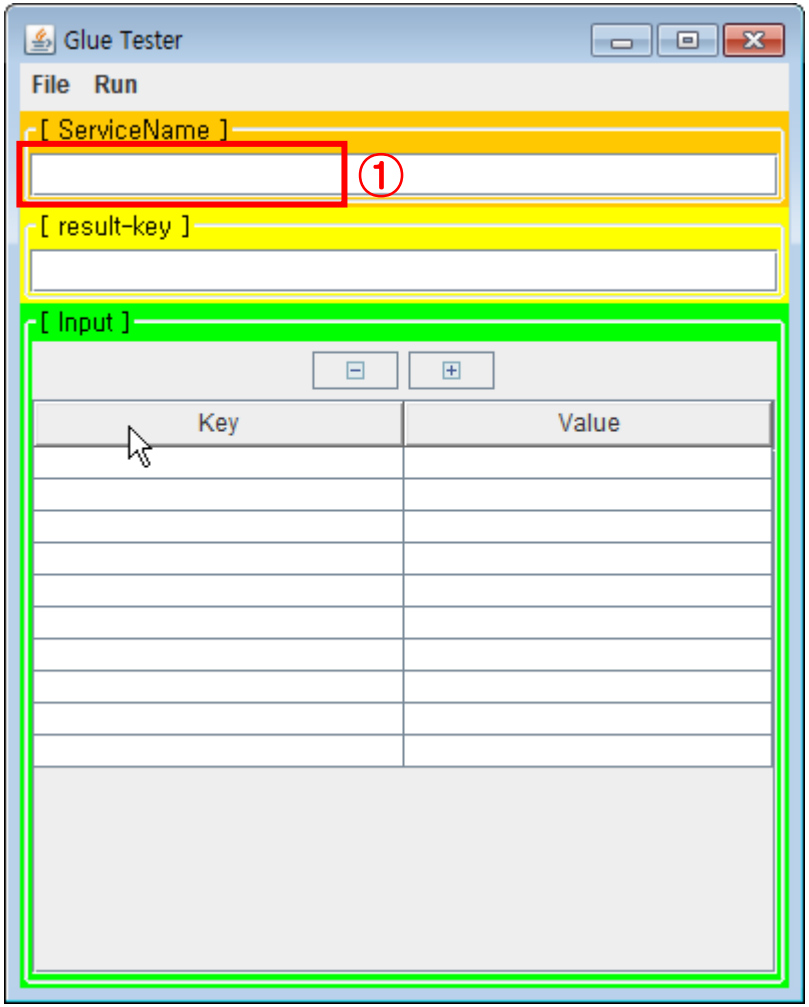
GlueTester실행

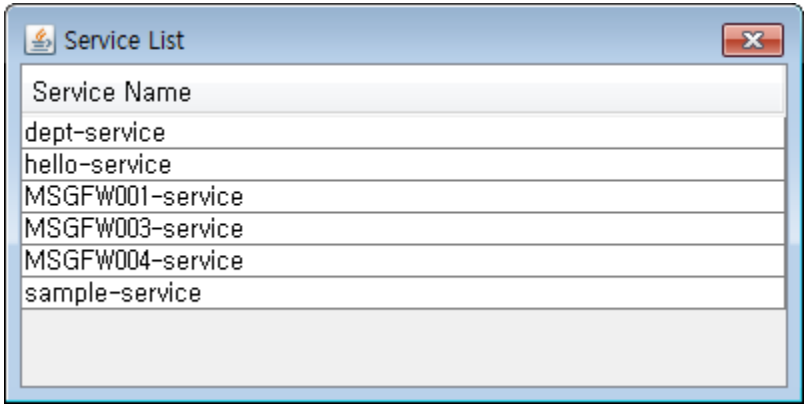
Run_GlueTester 로 실행할 경우



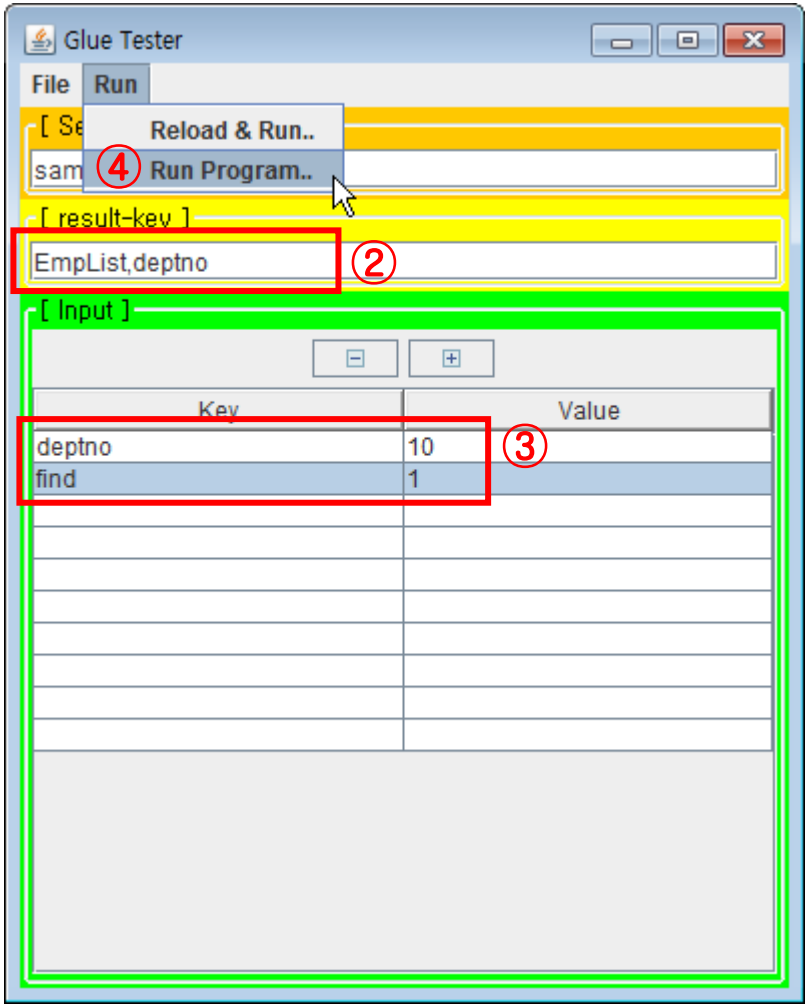
GlueTester실행

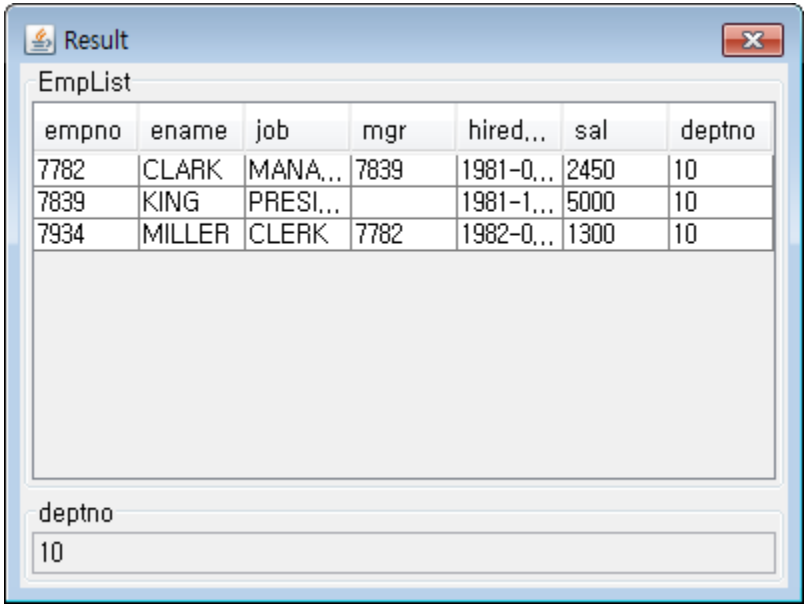
Glue Tester 및 Service List

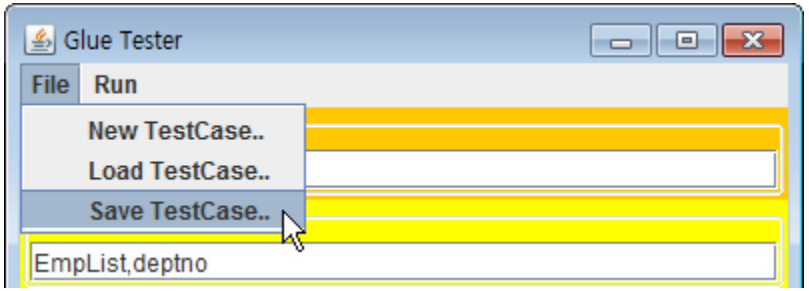




- GlueTester실행
  - 실행결과







Glue Framework 별첨

- **Bean 설정**
  - **messageSource를 Bean으로 등록**

```
<bean id="messageSource"
  class="org.springframework.context.support.ResourceBundleMessageSource">
  <property name="basenames">
    <list>
      <value>message</value>
    </list>
  </property>
</bean>
```

- **Property 파일 등록**
  - **messageSource에 등록된 이름 뒤에 Locale정보를 붙여서 Property 파일 생성**
    - 예: message_ko.properties, message_en.properties
    - 우리 나라와 같이 2byte 문자를 사용하는 경우는 MessageSource에 변경 사항이 발생할 때마다 native2ascii 명령어를 실행하여 파일을 다시 생성해 주어야 하는 번거로움이 있다. 그러나 ResourceBundleEditor 이클립스 플러그 인을 사용하면 쉽게 작성할 수 있다.

- **Locale 우선 순위**
  - **User 설정 Locale -> Glue Property 설정(default.msg.locale) -> System Locale**

Globalization 기능 이용

Java Util : GlueStaticContext.getResourceMessage(….)

- getResourceMessage(String beanName, String key, Object[] arguments, Locale locale)
  - beanName : ResourceBundleMessageSource가 등록된 Bean ID
  - key : property 파일에 정의된 message ID
  - arguments : Message에 바인딩 되는 값들
  - locale : 적용 Locale, Locale이 ko_KR인 경우 message_ko_KR.properties파일의 값을 읽어온다.
- getResourceMessage(key)
- getResourceMessage(key , locale)
- getResourceMessage(key, arguments)
- getResourceMessage(key, arguments, locale) :
  - Parameter로 Locale 이 없을 경우 선 순위
    - User 설정 Locale -> Glue Property 설정 Locale(default.ui.locale) -> System Locale
  - Parameter로 beanName 이 없을 경우 Default값
    - messageSource

사용예

```
String key = "showtable.header.empinfo";
System.out.println("Default::::"+GlueStaticContext.getResourceMessage(key));
System.out.println("English::::"+GlueStaticContext.getResourceMessage(key, Locale.US));
System.out.println("한국어::::"+GlueStaticContext.getResourceMessage(key, Locale.KOREA));
```

Multi Time Zone 기능 이용

Java Util : GlueStaticContext. getTimeZoneFormat(….)

- getTimeZoneFormat(Date date, String timezoneid, String textformat)
  - date : Text 형태로 변환 할 Date
  - timezoneid : Text 형태로 변환 시 적용 할 Time Zone ID
  - textformat : Text 형태로 변환 시 적용 될 Text Format (Default 값 : yyyy-MM-dd HH:mm:ss (z Z))

- getTimeZoneFormat(Date date, int offset, String textformat)
  - date : Text 형태로 변환 할 Date
  - offset : Text 형태로 변환 시 적용 할 Time Zone 의 OffSet값(분 단위 : 120일 경우 GMT +02:00)
  - textformat : Text 형태로 변환 시 적용 될 Text Format

사용예

```
Date date = new Date();
String textformat = "yyyy-MM-dd HH:mm:ss (z Z)";

System.out.println("Default::::"+GlueStaticContext.getTimeZoneFormat(date, TimeZone.getDefault().getID(), textformat));
System.out.println("Asia/Seoul::::"+GlueStaticContext.getTimeZoneFormat(date, "Asia/Seoul", textformat));
System.out.println("Canada/Eastern::::"+GlueStaticContext.getTimeZoneFormat(date, "Canada/Eastern", textformat));
System.out.println("Asia/Shanghai::::"+GlueStaticContext.getTimeZoneFormat(date, "Asia/Shanghai", textformat));
System.out.println("+05:00::::"+GlueStaticContext.getTimeZoneFormat(date, 5*60, textformat));
System.out.println("-06:00::::"+GlueStaticContext.getTimeZoneFormat(date, -6*60, textformat));
```

- **Lib**
  - **gluelib**
    - glue-schema-4.x.x.jar
      - Xml 스키마 기반의 파싱 처리 모듈
      - GlueService, GlueQueryEditor, LayoutDefinition
    - glue-core-4.x.x.jar
      - 개발 Application 의 실행 담당.
      - Logging, Caching, Controller, Manager 등 포함.
    - glue-tester-4.x.x.jar
      - Glue Tester 모듈
    - glue-hibernate-4.x.x.jar
      - Hibernate용 DAO, Hibernate용 Transaction Manager, Hibernate용 Reuse Activity
    - glue-mybatis-4.x.x.jar
      - Mybatis용 DAO, Mybatis용 Reuse Activity
    - glue-schedule-4.x.x.jar
      - Quartz, Spring Scheduler
    - glue-ws-4.x.x.jar
      - WebService Controller

- **Lib**
  - **gluestd**
    - XML Bean Library
      - xmlbeans-2.4.0.jar
      - jaxp-api-1.4.2.jar
    - Spring Framework Library
      - spring-aop-3.2.x.RELEASE.jar
      - spring-beans-3.2.x.RELEASE.jar
      - spring-context-3.2.x.RELEASE.jar
      - spring-context-support-3.2.x.RELEASE.jar
      - spring-core-3.2.x.RELEASE.jar
      - spring-expression-3.2.x.RELEASE.jar
      - spring-jdbc-3.2.x.RELEASE.jar
      - spring-orm-3.2.x.RELEASE.jar
      - spring-tx-3.2.x.RELEASE.jar
      - spring-web-3.2.x.RELEASE.jar
      - spring-webmvc-3.2.x.RELEASE.jar
      - spring-webmvc-portlet-3.2.x.RELEASE.jar
      - aopalliance-1.0.jar

→ 필수 : **XML Schema**기반의 **xml** 파일 파싱 시 사용. 둘 다 필수.

→ 필수 : **Spring Framework**을 활용/확장했으므로 필수.
단, **portlet, orm** 등은 선택사항임.

- **Lib**
  - **gluestd**
    - Logging Library
      - slf4j-api-1.7.2.jar
      - logback-core-1.0.13.jar
      - logback-classic-1.0.13.jar
      - jcl-over-slf4j-1.6.6.jar

      → 필수 : **slf4j-api**는 필수이나 그외는 **Log**구현체는 선택사항임.
    - Caching Library
      - ehcache-core-2.6.6.jar
      - jcs-1.3.jar

      → 필수 : **EhCache, JCS** 둘 중 하나 선택.
    - Common Lang 3
      - commons-lang3-3.1.jar

      → 필수
    - JDBC Driver Library
      - commons-dbcp-1.4.jar
      - commons-pool-1.6.jar
      - sqlite-jdbc-3.7.2.jar
      - ojdbc.jar

      → **DB** 연결시 필요.
    - Jason Library
      - jackson-core-asl-1.9.12.jar
      - jackson-mapper-asl-1.9.12.jar

      → **Jason Data** 처리시 필요.

Glue Framework

- **Lib**
  - **gluestd**
    - Apache CXF Library 2.7.3
    - Quartz Library 1.8.6
    - poi , jxl library
    - Hibernate Library 3.3.1
    - Mybatis Library 3.1.1.
    - Spring Mobile
    - Struts 1

Glue Master

- **Master Data**
  - **비즈니스 업무를 시스템으로 처리하기 위한**
  - **기본적인 기준정보를 통합 일원화 및 표준화한 데이터**
- **GlueMaster**
  - **GlueMaster Manager : Master Data 관리 도구**
  - **GlueMaster Access : Master Data Access API**
    - com.poscoict.app.glue.master.access.MasterAccess
    - com.poscoict.app.glue.master.access.type.PosCodeValueInfo
    - com.poscoict.app.glue.master.access.type.PosCodeValueList
    - com.poscoict.app.glue.master.access.type.PosRuleValueInfo
    - com.poscoict.app.glue.master.access.type.PosCalcValueInfo

Master API 사용 환경 구성

Library 추가

```
<dependency>
    <groupId>com.poscoict.app</groupId>
    <artifactId>glue-master-access</artifactId>
    <version>2.1.5-RELEASE</version>
</dependency>
<dependency>
    <groupId>org.apache.jcs</groupId>
    <artifactId>jcs</artifactId>
    <version>1.3</version>
    <exclusions>
        <exclusion>
            <artifactId>servlet-api</artifactId>
            <groupId>javax.servlet</groupId>
        </exclusion>
    </exclusions>
</dependency>
<dependency>
    <groupId>com.oracle</groupId>
    <artifactId>ojdbc6</artifactId>
    <version>1.0</version>
    <scope>system</scope>
    <systemPath>C:/eclipse/users/GlueSDK/repo/ext/ojdbc6.jar</systemPath>
</dependency>
```

→ Local Repository
(또는 Private Reposiitory)

→ Maven Central Repository

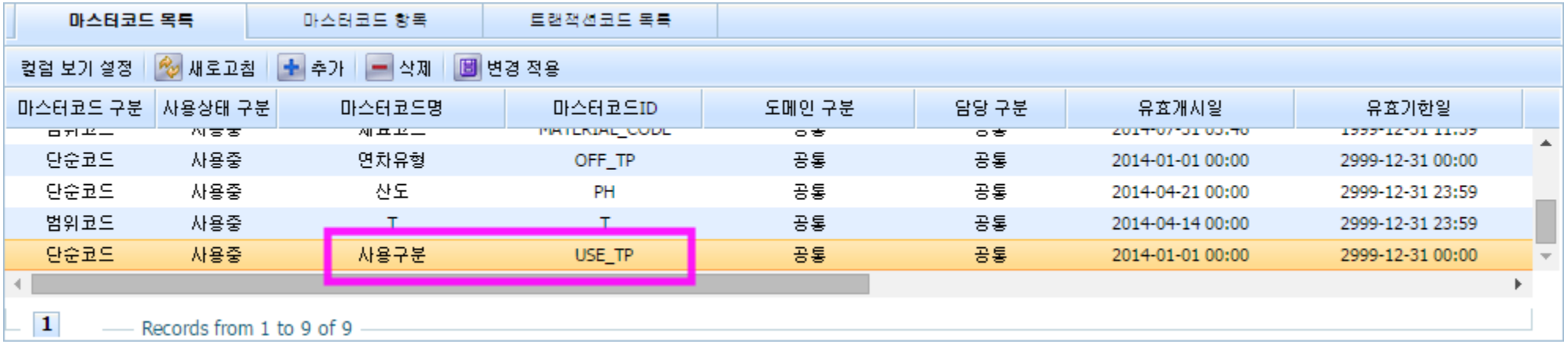
→ System Path 이용

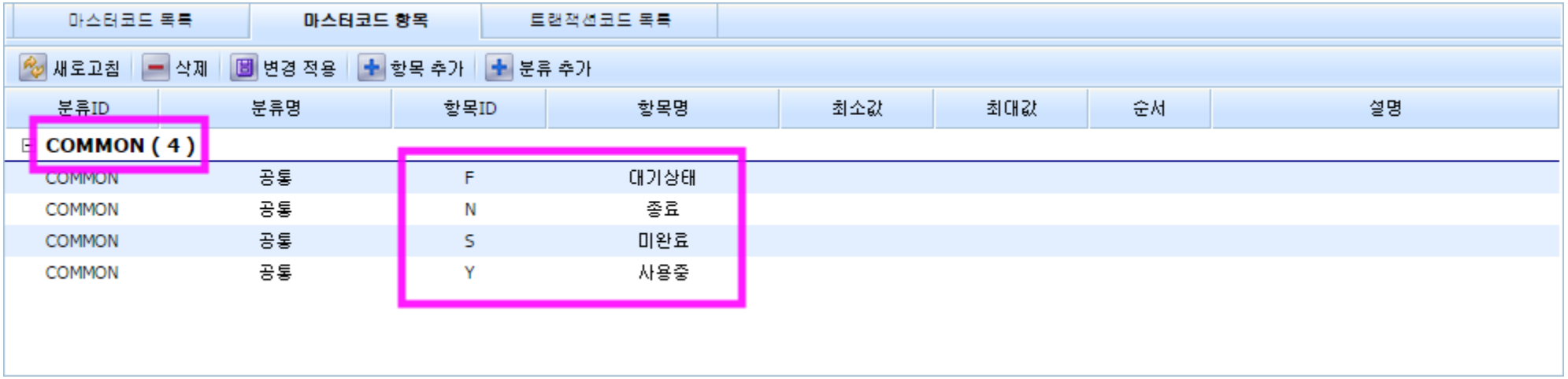
- **Master API 사용 환경 구성**
  - **GlueMaster.properties 추가**
    - 위치 : CONFIG_PATH 로 지정한 곳

```
access.datasource.name=accessDS
access.jdbc.driver=oracle.jdbc.driver.OracleDriver
access.jdbc.url=jdbc:oracle:thin:@192.168.41.141:1521:XE
access.jdbc.username=GM_USER
access.jdbc.password=GM_USER
access.querypath=access-query/oracle
access.rmi.registryPort=1199
```

마스터코드 예시

USE_TP





테스트

Test_MasterAccess_Code.java

```
import com.poscoict.app.glue.master.access.MasterAccess;
import com.poscoict.app.glue.master.access.type.PosCodeValueInfo;
import com.poscoict.app.glue.master.access.type.PosCodeValueList;

public class Test_MasterAccess_Code
{
    public static void main( String[] args )
    {
        System.setProperty( "CONFIG_PATH", "C://apache-tomcat-7.0.57" );
        PosCodeValueList codeList = MasterAccess.getCodeList( "USE_TP", "COMMON" );
        for ( PosCodeValueInfo code : codeList )
        {
            System.out.println(code.getCodeValue() + "/" + code.getCodeMean());
        }
        PosCodeValueInfo code = MasterAccess.getCodeValueInfo( "USE_TP", "COMMON", "Y" );
        System.out.println(code.getCodeMean());
        System.exit( 0 );
    }
}
```

테스트

Test_MasterAccess_Code.jsp (1/3)

```
<!DOCTYPE html>
<%@ page contentType="text/html;charset=utf-8"%>
<%@ page import="com.poscoict.app.glue.master.access.MasterAccess" %>
<%@ page import="com.poscoict.app.glue.master.access.type.*" %>
<html>
<head>
  <title>Master Access API 실습</title>
  <meta charset="utf-8">
  <style>
header, nav, section, article, footer {border:1px solid grey; margin:5px; padding:8px;}
nav ul {margin:0; padding:0;}
nav ul li{display:inline; margin:5px;}
  </style>
</head><%
  String codeId = request.getParameter("code-id")==null
                      ? "USE_TP" : request.getParameter("code-id");
  String categoryId = request.getParameter("category-id")==null
                      ? "COMMON" : request.getParameter("category-id");
  String error = null;
  PosCodeValueList codeList = null;
  try{
    codeList = MasterAccess.getCodeList( codeId, categoryId );
  }catch(Exception e){
    error = e.getMessage();
  }  %>
```

테스트

Test_MasterAccess_Code.jsp (2/3)

```
<body>
  <header>
    <h2>MasterAccess API Test</h2>
    <form>
      <input type=text name="code-id" value="<%=codeId%>">
      <input type=text name="category-id" value="<%=categoryId%>">
      <input type=submit>
    </form>
  </header>

  <nav>
    <ul>
      <li><a href="Test_MasterAccess_Code.jsp">마스터코드</a></li>
      <li><a href="Test_MasterAccess_Rule.jsp">업무기준</a></li>
      <li><a href="Test_MasterAccess_Calc.jsp">계산수식</a></li>
    </ul>
  </nav>

  <section>
    <h3>마스터코드 테스트 결과</h3><% if(codeList!=null){ %>
    <article>
      마스터코드 ID : <%= codeId %><br>
      분류 ID : <%= categoryId %>
    </article>
```

테스트

Test_MasterAccess_Code.jsp (3/3)

```
    <article>
      <table width=100%>
        <tr bgcolor=#CCEEBB>
          <td>코드</td>
          <td>의미</td>
        </tr><% for ( PosCodeValueInfo code : codeList ){ %>
        <tr bgcolor=#DDEEBB>
           <td><%= code.getCodeValue()%></td>
           <td><%= code.getCodeMean()%></td>
        </tr><% }%>
      </table>
    </article>
    <article>
      <%= categoryId %>에 해당하는 코드는 <%= codeList.size() %> 개 있습니다.
    </article><% }else{ %>
    <article>
      ERROR : <%= error %>
    </article><% } %>
  </section>
</body>
</html>
```

업무기준 예시

연필심구성성분

업무기준 목록 | 업무기준 레이아웃 | 업무기준 데이터

컬럼 보기 설정 | 새로고침 | 추가 | 삭제 | 변경 적용

업무기준 구분	사용상태 구분	업무기준명	업무기준ID	도메인 구분	담당 구분	유효개시일	유효기한일
단순 일반	사용중	에너지관리공정기준	FEQSP003	공통	공통	2013-09-06 00:00	2023-09-06 00:00
단순 일반	사용중	ISA95구조	ISA95MODEL	공통	공통	2013-08-20 00:00	2023-08-20 00:00
연관 판단	사용중	BMI(비만도)기준	RD#0001	공통	공통	2014-01-01 00:00	2999-12-31 00:00
단순 일반	사용중	연필심구성성분	RG#0001	공통	공통	2014-01-01 00:00	2999-12-31 00:00

1 Records from 1 to 10 of 10

업무기준 목록 | 업무기준 레이아웃 | 업무기준 데이터

새로고침 | 추가 | 삭제 | 변경 적용

순서	조건	결과		
	연필심번호	흑연	점토	왁스
0	예외처리	예외Data	예외Data	예외Data
1	9H	41	53	5
2	8H	44	50	5
3	7H	47	47	5
4	6H	50	45	5

테스트

Test_MasterAccess_Rule.java

```
import com.poscoict.app.glue.master.access.MasterAccess;
import com.poscoict.app.glue.master.access.type.PosRuleValueInfo;

public class Test_MasterAccess_Rule
{
    public static void main( String[] args )
    {
        System.setProperty( "CONFIG_PATH", "C://apache-tomcat-7.0.57" );
        String[] conditions = new String[]{"4B"};
        PosRuleValueInfo result = MasterAccess.getRuleValueInfo( "RG#0001", conditions );
        System.out.println( result );
        System.exit( 0 );
    }
}
```

테스트

Test_MasterAccess_Rule.jsp (1/3)

```
<!DOCTYPE html>
<%@ page contentType="text/html;charset=utf-8"%>
<%@ page import="java.util.*" %>
<%@ page import="com.poscoict.app.glue.master.access.MasterAccess" %>
<%@ page import="com.poscoict.app.glue.master.access.type.*" %>
<html>
<head>
  <title>Master Access API 실습</title>
  <meta charset="utf-8">
  <style>
header, nav, section, article, footer {border:1px solid grey; margin:5px; padding:8px;}
nav ul {margin:0; padding:0;}
nav ul li{display:inline; margin:5px;}
  </style>
</head><%

  String ruleId = request.getParameter("rule-id")==null
        ? "RG#0001" : request.getParameter("rule-id");
  String input = request.getParameter("rule-input")==null
        ? "" : request.getParameter("rule-input");
  String[] inputs = input.length()==0 ? new String[]{} : input.split(",");
  String error = null;
  PosRuleValueInfo result = null;
  List<String> ids = null;
```

테스트

Test_MasterAccess_Rule.jsp (1/3)

```
  try{
    result = MasterAccess.getRuleValueInfo( ruleId, inputs );
    ids = result.getItemIDList();
  }catch(Exception e){
    error = e.getMessage();
  }
%>
<body>
  <header>
    <h1>MasterAccess API Test</h1>
    <form>
      <input type=text name="rule-id" value="<%=ruleId%>">
      <input type=text name="rule-input" value="<%=input%>">
      <input type=submit>
    </form>
    업무기준ID와 조건은 컴마(,)로 구분해서 입력합니다.
  </header>
  <nav>
    <ul>
      <li><a href="Test_MasterAccess_Code.jsp">마스터코드</a></li>
      <li><a href="Test_MasterAccess_Rule.jsp">업무기준</a></li>
      <li><a href="Test_MasterAccess_Calc.jsp">계산수식</a></li>
    </ul>
  </nav>
```

테스트

Test_MasterAccess_Rule.jsp (1/3)

```
 <section>
    <h2>업무기준 테스트 결과</h2><% if(result!=null){ %>
    <article><pre><%= result.getHeaderInfo() %></pre></article>
    <article>
      <table width=100%>
        <tr bgcolor=#CCEEBB><% for ( String id : ids) { %>
          <td><%= id %></td><% }%>
        </tr><% for ( Map<String, String> rule : result ){ %>
        <tr bgcolor=#DDEEBB><% for ( String id : ids) { %>
          <td><%= rule.get(id) %></td><% }%>
        </tr><% }%>
      </table>
    </article><% }else{ %>
    <article>
      ERROR : <%= error %>
    </article><% } %>
  </section>
</body>
</html>
```

계산수식 예시

단리계산식

계산수식 목록 | 계산수식 입력항목 | 계산수식 공식

컬럼 보기 설정 | 새로고침 | 추가 | 삭제 | 변경 적용

계산수식 구분	사용상태 구분	계산수식명	계산수식ID	도메인 구분	담당 구분	유효개시일	유효기한일
단순 수식	사용중	단리계산	CAL#001	공통	공통	2015-01-01 00:00	2999-12-31 00:00
단순 수식	사용중	복리계산	CAL#002	공통	공통	2015-01-01 00:00	2999-12-31 00:00
판단 수식	사용중	근로소득공제액 계산	CAL#003	공통	공통	2015-01-01 00:00	2999-12-31 00:00

1 Records from 1 to 9 of 9

계산수식 목록 | 계산수식 입력항목 | 계산수식 공식

새로고침 | 추가 | 삭제 | 변경 적용

항목순서	표시	표준항목명	다국어항목명	표준항목ID	데이터구분	길이	단위	처리타입	상세입력	수정자	수정일시
1	V1	원금	원금	PRINCIPAL	Number	10	원	변수	-	edu	2015-02-09 17:33
2	V2	이자	이자	INTEREST	Number	2		변수	-	edu	2015-02-09 17:33
3	V3	기간	기간	PERIOD	Number	3	year	변수	-	edu	2015-02-09 17:33

계산수식 목록 | 계산수식 입력항목 | 계산수식 공식

새로고침 | 추가 | 삭제 | 변경 적용

공식순서	계산공식
1	V1*(1+V2*V3)

새로고침 | 삭제 | 변경 적용

Input 조건

계산식 조건: V1>0 and V2<1 and V3>0

테스트

Test_MasterAccess_Calc.java

```
import java.util.*;
import com.poscoict.app.glue.master.access.MasterAccess;
import com.poscoict.app.glue.master.access.type.PosCalcValueInfo;

public class Test_MasterAccess_Calc
{
    public static void main( String[] args )
    {
        System.setProperty( "CONFIG_PATH", "C://apache-tomcat-7.0.57" );
        List<String[]> values = new ArrayList<String[]>();
        values.add( new String[] { "1000000" } );
        values.add( new String[] { "0.1" } );
        values.add( new String[] { "3" } );
        values.add( new String[] { "2" } );
        PosCalcValueInfo result = MasterAccess.getCalcValueInfo( "CAL#002", values );
        System.out.println( result );;
        System.exit( 0 );
    }
}
```

테스트

Test_MasterAccess_Calc.jsp (1/3)

```
<!DOCTYPE html>
<%@ page contentType="text/html;charset=utf-8"%>
<%@ page import="java.util.*" %>
<%@ page import="com.poscoict.app.glue.master.access.MasterAccess" %>
<%@ page import="com.poscoict.app.glue.master.access.type.*" %>
<html>
<head>
  <title>Master Access API 실습</title>
  <meta charset="utf-8">
  <style>
header, nav, section, article, footer {border:1px solid grey; margin:5px; padding:8px;}
nav ul {margin:0; padding:0;}
nav ul li{display:inline; margin:5px;}
  </style>
</head><%
  String calcId = request.getParameter("calc-id")==null
      ? "CAL#001" : request.getParameter("calc-id");
  String input = request.getParameter("calc-input")==null
      ? "1000000/0.1/3" : request.getParameter("calc-input");
  String[] inputs = input.length()==0 ? new String[]{} : input.split("/");
  List<String[]> inputList = new ArrayList<String[]>();
  for( String ins : inputs ){
      inputList.add(ins.split( "," ));
  }
```

테스트

Test_MasterAccess_Calc.jsp (1/3)

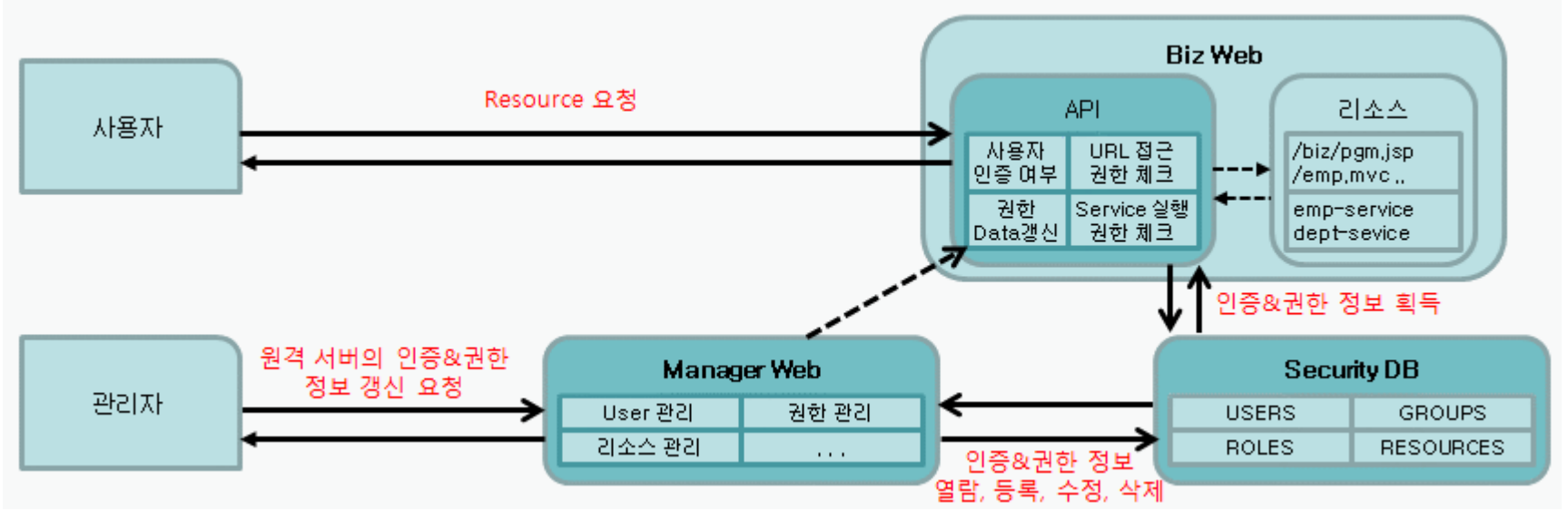
```
  String error = null;
  PosCalcValueInfo result = null;
  try{
    result = MasterAccess.getCalcValueInfo( calcId, inputList );
  }catch(Exception e){
    error = e.getMessage();
  }
%>
<body>
  <header>
    <h1>MasterAccess API Test</h1>
    <form>
      <input type=text name="calc-id" value="<%=calcId%>">
      <input type=text name="calc-input" value="<%=input%>">
      <input type=submit>
    </form>
    계산수식ID와 조건은 슬래시(/)와 컴마(,)로 구분해서 입력합니다.
  </header>
  <nav>
    <ul>
      <li><a href="Test_MasterAccess_Code.jsp">마스터코드</a></li>
      <li><a href="Test_MasterAccess_Rule.jsp">업무기준</a></li>
      <li><a href="Test_MasterAccess_Calc.jsp">계산수식</a></li>
    </ul>
  </nav>
```

테스트

Test_MasterAccess_Calc.jsp (1/3)

```
 <section>
    <h2>계산수식 테스트 결과</h2><% if(result!=null){ %>
    <article><pre><%= result.getHeaderInfo() %></pre></article>
    <article>
      <%= result.getFormula() %> =>
      <b><font color=red><%= result.getCalcResultValue() %></font></b>
    </article>
    <article>
      <table width=100%>
        <tr bgcolor=#CCEEBB>
          <td>변수</td>
          <td>값</td>
        </tr><% for ( String key : result.getInputValues().keySet() ){ %>
        <tr bgcolor=#DDEEBB>
           <td><%= key %></td>
           <td><%= result.getInputValues().get( key ) %></td>
        </tr><% }%>
      </table>
    </article><% }else{ %>
    <article>ERROR : <%= error %></article><% } %>
  </section>
</body>
</html>
```

- **Security Data**
  - **Web Application을 위한 사용자 인증 및 권한 관리**
  - **Authorization(권한)**
    - 웹에서 제공하는 자원(웹페이지, 이미지, 사운드, 동영상 등등)을 접근할 자격이 있는지
  - **Authentication(인증)**
    - 권한을 결정하기 전에 사용자가 Application(사이트)을 이용할 자격이 있는지



Glue Security [실습]

- **API 적용 절차**
  - **라이브러리 추가**
    - glue-security-access
  - **web.xml 수정**
    - filter, listener, servlet 추가
  - **security-context.xml, security-servlet.xml 추가**
  - **login.jsp, denied.jsp, menu.jsp 추가**
  - **applicationContext.xml 수정**
    - securityDS, securityTx, securityDao, viewAuthorityManager, serviceAuthorityManager 추가
  - **security-service.xml, security-query.glue_sql 추가**

- **http://www.solutionpot.co.kr/doc/security 에서 API 적용 가이드 참고**

- 리소스 유형(권한 체크)
  - SERVICE
  - URL

```
/sample.mvc
/sample_vo.mvc
/sample_global.mvc
/dept.mvc
/dept_jquery.mvc
/gm/Test_MasterAccess_Calc.jsp
/gm/Test_MasterAccess_Code.jsp
/gm/Test_MasterAccess_Calc.jsp
/check.jsp
/springmvc/sample.jsp
/springmvc/sample_vo.jsp
/springmvc/sample_global.jsp
/springmvc/dept.jsp
/springmvc/dept_jquery.jsp
/ws/glueJaxWs
```

```
dept-service
sample-service
sample01-service
sample02-service
hello-service
MSGFW001-service
MSGFW003-service
MSGFW004-service
```

Security API 사용 환경 구성

Library 추가

```
<dependency>
    <groupId>com.poscoict.app</groupId>
    <artifactId>glue-security-access</artifactId>
    <version>3.0.3-RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>3.2.4.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>3.2.4.RELEASE</version>
</dependency>
<dependency>
    <groupId>jstl</groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
```

→ Local Repository
(또는 Private Repository)

→ Maven Central Repository

- Oracle JDBC Driver
- Jackson-Mapper

Security API 사용 환경 구성

web.xml 에 다음 추가 (1/2)

```
<web-app . . .>

    <listener>
        <listener-class>
           org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
            /WEB-INF/jaxws-server-endpoint.xml
            /WEB-INF/security/security-context.xml
        </param-value>
    </context-param>

    <filter>
        <filter-name>springSecurityFilterChain</filter-name>
        <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>springSecurityFilterChain</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
```

Security API 사용 환경 구성

web.xml 에 다음 추가 (2/2)

```
    <servlet>
        <servlet-name>security</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>security</servlet-name>
        <url-pattern>/SecurityReload</url-pattern>
    </servlet-mapping>

</web-app>
```

- security-context.xml은 http://www.solutionpot.co.kr/doc/security 참고
- security-servlet.xml은 http://www.solutionpot.co.kr/doc/security 참고

Security API 사용 환경 구성

security-context.xml 추가

```
<beans:beans . . .>

    <http auto-config="true" access-denied-page="/denied.jsp">
        <custom-filter ref="filterInvocationInterceptor"
                       after="FILTER_SECURITY_INTERCEPTOR" />
        <custom-filter ref="gluefilter"
                       before="FILTER_SECURITY_INTERCEPTOR" />
        <form-login login-page="/login.jsp"
                default-target-url="/index.jsp"
                login-processing-url="/j_spring_security_check"
                username-parameter="j_username"
                password-parameter="j_password"
                always-use-default-target="true" />
        <logout logout-url="/j_spring_security_logout"
            logout-success-url="/index.jsp"
            invalidate-session="true" delete-cookies="JSESSIONID"/>
    </http>

. . .
</beans:beans>
```

- login.jsp와 denied.jsp는 http://www.solutionpot.co.kr/doc/security 참고

Security API 사용 환경 구성

index.jsp (1/2)

```
<%@ page contentType="text/html;charset=utf-8"%>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Menu</title>
  <style>
header, nav, section, article, footer {border:1px solid grey; margin:5px; padding:8px;}
nav ul {margin:0; padding:0;}
nav ul li{display:inline; margin:5px;}
  </style>
</head>
<body>
  <header><%
    if(session.getAttribute("SPRING_SECURITY_CONTEXT") !=null ){ %>
    <c:if test="${pageContext.request.userPrincipal.name!=null}">
    Welcome <b>${pageContext.request.userPrincipal.name}</b>.
    </c:if>
    <div align=right><a href="j_spring_security_logout">logout</a></div><%
   }else{ %>
    <img src="http://www.solutionpot.co.kr/doc/framework/images/bannerLeft.gif"><%
   } %>
  </header>
```

Security API 사용 환경 구성

index.jsp (1/2)

```
  <nav>
    <ul>
      <li><a href="./sample.mvc">사원정보(List&lt;Map&gt;)</a></li>
      <li><a href="./sample_vo.mvc">사원정보(List&lt;EmpVO&gt;)</a></li>
      <li><a href="./sample_global.mvc">사원정보(다국어)</a></li>
    </ul>
  </nav>
  <nav>
    <ul>
      <li><a href="./dept.mvc">부서정보</a></li>
      <li><a href="./dept_jquery.mvc">부서정보(jquery)</a></li>
    </ul>
  </nav>
  <nav>
    <ul>
      <li><a href="./gm/Test_MasterAccess_Code.jsp">마스터코드</a></li>
      <li><a href="./gm/Test_MasterAccess_Rule.jsp">업무기준</a></li>
      <li><a href="./gm/Test_MasterAccess_Calc.jsp">계산수식</a></li>
    </ul>
  </nav>
</body>
</html>
```

Security API 사용 환경 구성

applicationContext.xml 에 다음 추가 (1/2)

```
<beans . . .>

    <bean id="securityDs" class="org.apache.commons.dbcp.BasicDataSource"
          destroy-method="close">
        <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/>
        <property name="url" value="jdbc:oracle:thin:@192.168.41.141:1521:XE"/>
        <property name="username" value="GLUESECURITY3"/>
        <property name="password" value="GLUESECURITY3"/>
        <property name="defaultAutoCommit" value="false"/>
        <property name="minIdle" value="0"/>
        <property name="maxActive" value="-1"/>
        <property name="maxIdle" value="1000"/>
    </bean>
    <bean id="securityTx"
          class="com.poscoict.glueframework.transaction.GlueDataSourceTransactionManager">
        <property name="dataSource" ref="securityDs"/>
    </bean>
    <bean id="securityDao" class="com.poscoict.glueframework.dao.jdbc.GlueJdbcDao">
        <property name="dataSource" ref="securityDs"/>
        <property name="queryManager" ref="queryManager"/>
    </bean>
```

Security API 사용 환경 구성

applicationContext.xml 에 다음 추가 (2/2)

```
    <bean id="viewAuthorityManager"
      class="com.poscoict.glueframework.security.authority.GlueViewAuthorityDbManagerImpl">
        <property name="jdbcDao" ref="securityDao"/>
    </bean>
    <bean id="serviceAuthorityManager"
   class="com.poscoict.glueframework.security.authority.GlueServiceAuthorityDbManagerImpl">
        <property name="jdbcDao" ref="securityDao"/>
    </bean>

. . .
</beans>
```

Security API 사용 환경 구성

security-service.xml 추가

```
<beans . . .>

    <bean id="serviceManager" . . ./>
    <bean id="cacheManager" . . ./>
    <bean id="serviceLoader"
          class="com.poscoict.glueframework.biz.control.GlueServiceLoader">
        <property name="extraServiceFiles">
            <list>
                <value>ref/security-service.xml</value>
            </list>
        </property>
    </bean>

. . .
</beans>
```

Security API 사용 환경 구성

security-query.glue_sql 추가

```
<beans . . .>

    <bean id="serviceManager" . . ./>
    <bean id="cacheManager" . . ./>
    <bean id="serviceLoader" . . ./>
    <bean id="queryManager" . . ./>
    <bean id="queryLoader" class="com.poscoict.glueframework.dao.manager.GlueQueryLoader">
        <property name="extraQueryFiles">
            <list>
                <value>ref/security-query.glue_sql</value>
            </list>
        </property>
    </bean>

. . .
</beans>
```