WINI Framework 가이드

시그니춰_좌우조합

개 정 이 력

| 버전 | 작성일 | 변경내용[[1]](#footnote-1) | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 1.0 | 2019.09.10 | 최초작성 | 최재영 | 곽주영 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목 차

[1 WINI Framework(위니 프레임워크) 5](#_Toc22717046)

[1.1 개요 5](#_Toc22717047)

[1.2 구조 5](#_Toc22717048)

[1.3 개발환경 6](#_Toc22717052)

[1.4 사용방법 7](#_Toc22717053)

[1.4.1 환경셋팅 7](#_Toc22717054)

[1.4.2 DB접속 변경 방법 8](#_Toc22717055)

[1.5 개발방법 10](#_Toc22717056)

[1.5.1 개발 규칙 10](#_Toc22717057)

[1.5.2 참고 소스(winiWeb 사용자관리 화면 조회, 등록, 수정, 삭제) 12](#_Toc22717058)

[2 WINI Component (Java) 13](#_Toc22717059)

[2.1 WiniService.java 13](#_Toc22717060)

[2.1.1 insert(insert SQL문 실행) 14](#_Toc22717061)

[2.1.2 update (update SQL문 실행) 15](#_Toc22717062)

[2.1.3 delete (delete SQL문 실행) 16](#_Toc22717063)

[2.1.4 selectByPk (단건 데이터 조회) 17](#_Toc22717064)

[2.1.5 list (jqgrid 사용시) 18](#_Toc22717065)

[2.1.6 list (jqgrid 사용시 paging포함) 19](#_Toc22717066)

[2.1.7 list\_json (winigrid 사용 시) 20](#_Toc22717067)

[2.1.8 list\_json (winigrid 사용 시 panging 포함) 21](#_Toc22717068)

[2.1.9 list\_easyui (easyuigrid 사용 시) 22](#_Toc22717069)

[2.1.10 list\_easyui (easyuigrid 사용 시 panging 포함) 23](#_Toc22717070)

[2.1.11 mult\_save (일괄 Service 처리) 24](#_Toc22717071)

[2.1.12 list\_map (리스트 조회값을 Map에 담아 반환) 26](#_Toc22717072)

[2.1.13 list\_map (List 형 반환 시 사용) 27](#_Toc22717073)

[2.1.14 detail\_map (List 반환 값을 Map에 담아 사용(count X)) 28](#_Toc22717074)

[2.1.15 list\_map\_json (list를 json형태로 반환 시 사용) 29](#_Toc22717075)

[3 WINI Compoment (Js) 30](#_Toc22717076)

[3.1 wini\_cal.js 30](#_Toc22717077)

[3.1.1 wini\_com\_set\_detepick (INPUT BOX달력 UI 셋팅) 30](#_Toc22717078)

[3.1.2 wini\_com\_set\_datepickStartEnd (시작일, 종료일 달력) 32](#_Toc22717079)

[3.1.3 wini\_cal\_setValue (INPUT BOX 날짜 셋팅) 33](#_Toc22717080)

[3.2 wini\_code.js 34](#_Toc22717081)

[3.2.1 CodeUtil.setCodeList (코드그룹을 이용한 최대 2단계 SELECT BOX구현) 34](#_Toc22717082)

[3.2.2 CodeUtil.setCodeList (라디오버튼 셋팅) 36](#_Toc22717083)

[3.3 wini\_default.js 37](#_Toc22717084)

[3.3.1 wini\_getAjax 38](#_Toc22717085)

[3.3.2 wini\_submit\_ingView (진행중 표시) 39](#_Toc22717086)

[3.3.3 MM\_moveURL (강제 화면 이동) 40](#_Toc22717087)

[3.3.4 zoom\_it (화면 확대/축소) 41](#_Toc22717088)

[3.3.5 checkBrowser (브라우저 버전) 42](#_Toc22717089)

[3.3.6 gfn\_check\_tel (전화번호 유효성 체크) 43](#_Toc22717090)

[3.3.7 gfn\_check\_tel2 (전화번호 유효성 체크2) 44](#_Toc22717091)

[3.3.8 gfn\_CheckValue (영문 & 숫자 입력 체크) 45](#_Toc22717092)

[3.3.9 emailCheck (email 유효성체크) 46](#_Toc22717093)

[3.3.10 gfn\_checkMobileScreen (스마트폰 유무 체크) 47](#_Toc22717094)

[3.3.11 gfn\_isNum (숫자만 체크) 48](#_Toc22717095)

[3.3.12 password\_check (비밀번호 유효성 체크) 49](#_Toc22717096)

[3.3.13 gfn\_check\_number (값 입력시 정수, 실수 제한) 50](#_Toc22717097)

[3.3.14 gfn\_extChk (파일 확장자 체크) 51](#_Toc22717098)

[3.3.15 gfn\_openPop (진행중 표시) 52](#_Toc22717099)

[3.3.16 wini\_myDateChk (입력된 날짜 유효성 체크) 53](#_Toc22717100)

[3.4 wini\_gird.js 54](#_Toc22717101)

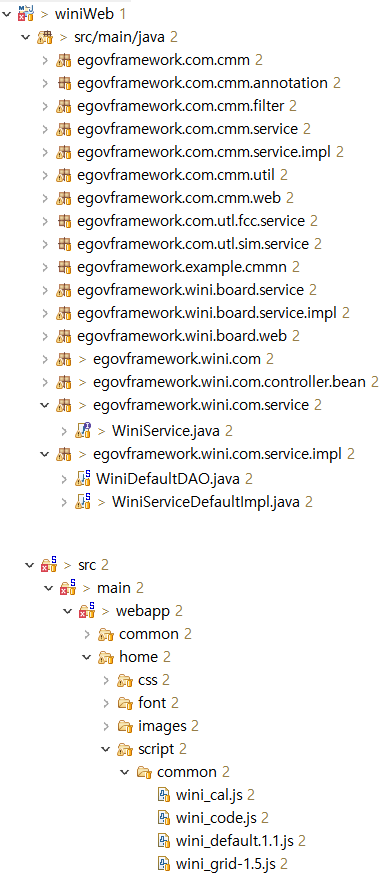
[3.4.1 winigrid 그리는 법 54](#_Toc22717102)

[3.4.2 grid\_detail\_Info (그리드 목록 클릭) 57](#_Toc22717103)

1. WINI Framework(위니 프레임워크)
   1. 개요

신규 인력에 대한 개발표준 교육과 다수의 프로젝트에 개발표준을 적용하여 필요한 기능을 미리 만들어 제공하여 개발 효율성, 생산성 및 코드 품질을 향상시킨다.

* 1. 구조



**WINI Framework**

WINI Framework는 전자정부 프레임워크를 기반으로JAVASCRIPT, Service, ServiceImpl, DAO 로 구성되어 있다.

전자정부 프레임워크

: 공공사업에 적용되는 개발프레임워크의 표준 정립으로 정보시스템 개발을 위해 필요한 기능 및 아키텍처를 제공하여 응용 SW 표준화, 품질 및 재 사용성 향상 등 효율적인 어플리케이션 구축을 위한 프레임워크이다.

※ 표준프레임워크는 기존 다양한 플랫폼(.NET, php 등) 환경을 대체하기 위한 표준은 아니며, java 기반의 정보시스템 구축에 활용하실 수 있는 개발·운영 표준 환경을 제공하기 위한 것입니다.

※ 사용하는 공통컴포넌트

|  |  |
| --- | --- |
| 패키지명 | 설명 |
| egovframework.com.cmm | 공통서비스 |
| egovframework.com.cmm.annotation | 공통컴포넌트 어노테이션 패키지 |
| egovframework.com.cmm.exception | 공통컴포넌트 예외사항 패키지 |
| egovframework.com.cmm.filter | 공통컴포넌트 필터 패키지 |
| egovframework.com.cmm.service | 공통컴포넌트 Service Interface, Model, VO패키지 |
| egovframework.com.cmm.service.impl | 공통컴포넌트 Service Impl, DAO 패키지 |
| egovframework.com.cmm.util | 공통컴포넌트 유틸리티 패키지 |
| egovframework.com.cmm.web | 공통컴포넌트 웹 패키지 |
| egovframework.com.utl.fcc.service | 공통컴포넌트 유틸리티 포맷/계산/변환 패키지 |
| egovframework.com.utl.sim.service | 공통컴포넌트 유틸리티 클라이언트 정보 패키지 |

전자정부 프레임워크에 관해서는 아래 주소를 참고한다.

표준프레임워크 소개

: <http://www.egovframe.go.kr/EgovIntro.jsp?menu=1&submenu=1>

전자정부 프레임워크 개발환경 3.8 가이드

: <https://www.egovframe.go.kr/wiki/doku.php?id=egovframework:dev3.8:dev3.8>

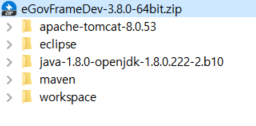
전자정부 프레임워크 온라인영상 교육

: <http://www.egovframe.go.kr/EgovEduMovie.jsp?menu=4&submenu=3>

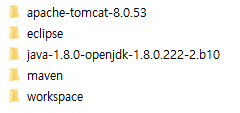
1. 3. 개발환경

|  |  |
| --- | --- |
| 항목 | 버전 |
| 전자정부 프레임워크 | egovframework 3.8  (Eclipse IDE for Java EE Developers 4.7.3 – Oxygen.3) |
| WAS | Apache Tomcat 8.0.53 |
| JDK | Openjdk java-1.8.0 |
| DBMS | Oracle 11g |
| Spring | Spring Framework 4.3.16 |
| MyBatis | 3.4.1 |

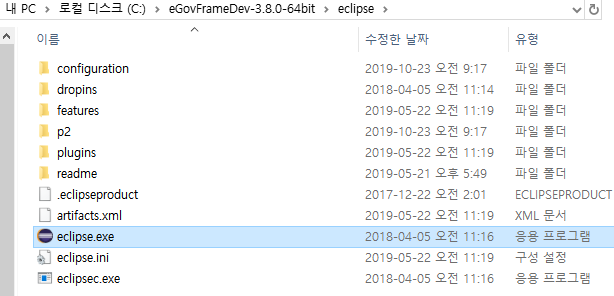
* 1. 사용방법
     1. 환경셋팅



전달받은 eGovFrameDev-3.8.0-64bit.zip파일을 C:\ 최상위에 압축 해제한다.



구성은 tomcat, eclipse, open jdk1.8, maven, workspace 로 되어 있다.

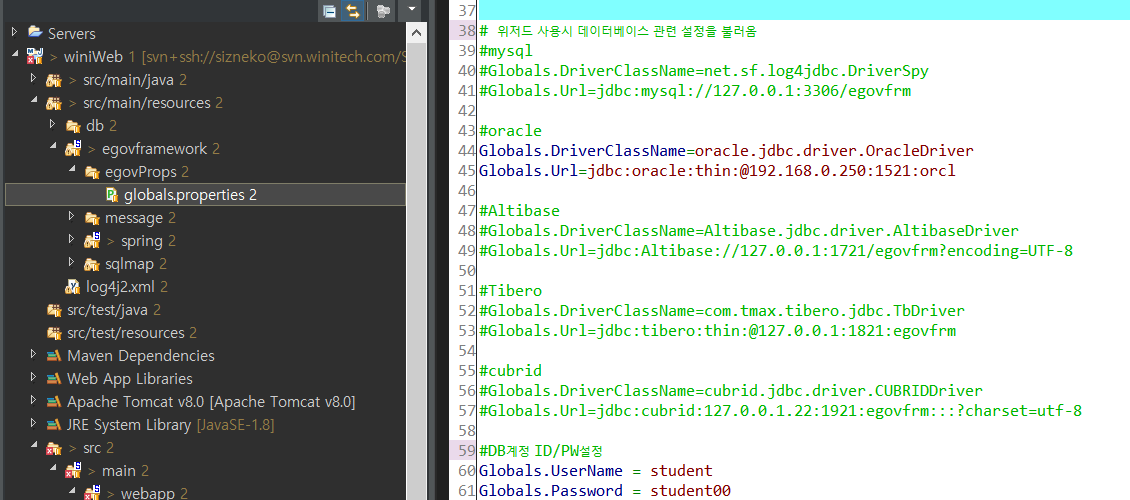


압축해제한 C:\eGovFrameDev-3.8.0-64bit\eclipse 경로에 eclipse.exe 를 실행하면 winiWeb 프로젝트가 실행된다.

* + 1. DB접속 변경 방법



1. winiWeb/src/main/resources/egovframework/spring/context-datasource.xml에 해당하는 DB(oracle, tibero, mysql 등)로 변경한다.



1. winiWeb/resources/egovframeework/egovProps/globals.properties 에

# 위저드 사용시 데이터베이스 관련 설정을 불러옴

위 주석의 위치로 이동하면 DB 접속 설정이 확인 가능하다.

1. Globals.DriverClassName, Globals.DriverClassName 을 수정하면 DB 드라이브와 접속 정보가 변경 가능하다.

Ex) 오라클

# 오라클 드라이버

Globals.DriverClassName=oracle.jdbc.driver.OracleDriver

# 오라클 접속 정보

Globals.Url=jdbc:oracle:thin:@192.168.0.250:1521:orcl

1. Globals.UserName, Globals.Password 을 수정하면 DB 접속 아이디, 비밀번호가 변경 가능하다.

Ex) # 오라클 아이디

Globals.UserName = student

# 오라클 비밀번호

Globals.Password = student00

* 1. 개발방법
     1. 개발 규칙

1. 화면 개발과 관련된 JAVA 추가시 특별한 기능을 제외한 화면기준으로 아래 규칙대로 생성한다.

폴더 생성시 해당 메뉴의 1차메뉴 명을 사용한다.

jsp, java, js 생성시 화면의 기능에 따라 생성한다.

// ex) 시스템관리 메뉴의 경우

// 폴더 생성 및 화면별 JSP 생성

winiWeb/src/main/webapp/WEB-INF/jsp/000

* 000Manager.jsp, 000List.jsp 등

ex) winiWeb/src/main/webapp/WEB-INF/jsp/sys/userManager.jsp

// 폴더 생성 및 화면별 JAVASCRIPT 생성

winiWeb/src/main/webapp/home/script/000

* 000Manager.js, 000List.js 등

ex) winiWeb/src/main/webapp/home/script/sys/userManager.js

// 화면 호출 Controller 가 포함된 package

winiWeb/src/main/java/egovframework/wini/000/web

* 000Controller.java

ex) winiWeb/src/main/java/egovframework/wini/sys/web/userController.java

// Service 및 VO 가 포함된 package

winiWeb/src/main/java/egovframework/wini/sys/service

* 000Service.java, 000VO.java

ex) winiWeb/src/main/java/egovframework/wini/sys/service/userService.java

winiWeb/src/main/java/egovframework/wini/sys/service/userVO.java

// ServiceImpl포함된 package

winiWeb/src/main/java/egovframework/wini/000/service/impl

* 000ServiceImpl.java, 000DAO

ex) /src/main/java/egovframework/wini/sys/service/impl/userServiceImpl.java

/src/main/java/egovframework/wini/sys/service/impl/userDAO

// xml mappig config

winiWeb/src/main/resources/egovframework/sqlmap/config

* sql-map-config-000.xml

ex) winiWeb/src/main/resources/egovframework/sqlmap/config

/sql-map-config-sys.xml

// sql xml

winiWeb/src/main/resources/ egovframework/sqlmap/프로젝트명/000

* 000\_SQL.xml, 000Manager\_SQL.xml 등

ex) winiWeb/src/main/resources/ egovframework/sqlmap/winiWeb/sys

/userManager\_SQL.xml

* + 1. 참고 소스(winiWeb 사용자관리 화면 조회, 등록, 수정, 삭제)
* src/main/webapp/WEB-INF/jsp/sys/userManager.jsp
* src/main/webapp/home/script/sys/userManager.js
* egovframework.wini.sys.web.UserManagerController.java
* src/main/resources/egovframework/sqlmap/config/sql-map-config-sys.xml
* src/main/resources/egovframework/sqlmap/wini/sys/userManager\_SQL.xml

1. WINI Component (Java)
   1. WiniService.java

|  |  |
| --- | --- |
| 위치 | /winiWeb/src/main/java/egovframework/wini/com/service |
| 설명 | SQL mapping 쿼리 ID와 입력 데이터를 셋팅한 파라메터 객체를 이용하여 SQL문을 실행시키는 Service |
| 필수 선언 | **import** java.util.List;  **import** java.util.Map;  **import** net.sf.json.JSONObject;  **import** egovframework.wini.com.controller.bean.easyuigridJason;  **import** egovframework.wini.com.controller.bean.jqgridJason; |

|  |  |
| --- | --- |
| 함수 | 설명 |
| insert | INSERT SQL문 실행 |
| update | UPDATE SQL문 실행 |
| delete | DELETE SQL문 실행 |
| selecByPk | 단건 데이터 조회 |
| list | JQGRID 활용 시 사용 |
| list | JQGRID 활용 시 사용(PAGING포함) |
| list\_json | WINIGRID 활용 시 사용 |
| list\_json | WINIGRID 활용 시(PANGING 포함) |
| list\_easyui | EASYUIGRID 활용 시 사용 |
| list\_easyui | EASYUIGRID 활용시 사용(PANGING 포함) |
| mult\_save | 일괄 SERVICE 처리 |
| list\_map | 리스트 조회값을 MAP에 담아 반환 |
| list\_map | LIST 형 반환 시 사용 |
| detail\_Map | LIST 반환 값을 MAP에 담아 사용(List COUNT X) |
| list\_map\_json | LIST를 JSON형태로 반환 시 사용 |

* + 1. insert(insert SQL문 실행)

|  |  |
| --- | --- |
| 원형 | Object insert(String queryId, Map param) **throws** Exception; |
| 설명 | 자주 사용 되는 SQL INSERT 문을 SQL커넥션을 따로 구현 하지 않고 INSERT문을 실행 할 수 있는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  **public** **void** insertStore(@ModelAttribute("searchVO") fwtVO fwtVO  ) **throws** Exception {  Map<String, Object> map1 = commandMap;  // 화면에서 전달한 값  map1.put("srch\_ward", commandMap.get("srch\_ward"));  winiService.insert("fwtStoreManager.INSERT\_HYD\_LGCY\_0001",map1);  } |
| 매개변수 | - queryId : 입력 처리 SQL mapping 쿼리 ID  - param : 입력 처리 SQL mapping 입력 데이터를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | 입력 시 selectKey 를 사용하여 key 를 딴 경우 해당 key |

* + 1. update (update SQL문 실행)

|  |  |
| --- | --- |
| 원형 | **int** update(String queryId, Map param) **throws** Exception; |
| 설명 | 자주 사용 되는 SQL INSET 문을 SQL커넥션을 따로 구현 하지 않고 UPDATE문을 실행 할 수 있는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  **public** **void** updateStore(@ModelAttribute("searchVO") fwtVO fwtVO  ) **throws** Exception {  Map<String, Object> map1 = commandMap;  // 화면에서 전달한 값  map1.put("srch\_ward", commandMap.get("srch\_ward"));  winiService.update("fwtStoreManager.UPDATE\_HYD\_LGCY\_0001",map1);  } |
| 매개변수 | - queryId : 수정 처리 SQL mapping 쿼리 ID  - param : 수정 처리 SQL mapping 입력 데이터(key 조건 및 변경 데이터)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | DBMS가 지원하는 경우 update 적용 결과 count |

* + 1. delete (delete SQL문 실행)

|  |  |
| --- | --- |
| 원형 | **int** delete(String queryId, Map param) **throws** Exception; |
| 설명 | 자주 사용 되는 SQL INSET 문을 SQL커넥션을 따로 구현 하지 않고 DELETE문을 실행 할 수 있는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  **public** **void** updateStore(@ModelAttribute("searchVO") fwtVO fwtVO  ) **throws** Exception {  Map<String, Object> map1 = commandMap;  // 화면에서 전달한 값  map1.put("srch\_ward", commandMap.get("srch\_ward"));  winiService.delete("fwtStoreManager.DELETE\_HYD\_LGCY\_0001",map1);  } |
| 매개변수 | - queryId : 삭제 처리 SQL mapping 쿼리 ID  - param : 삭제 처리 SQL mapping 입력 데이터(일반적으로 key 조건)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | DBMS가 지원하는 경우 delete 적용 결과 count |

* + 1. selectByPk (단건 데이터 조회)

|  |  |
| --- | --- |
| 원형 | Map<String, String> selectByPk(String queryId, Map param) **throws** Exception; |
| 기능 | 자주 사용 되는 SQL INSET 문을 SQL커넥션을 따로 구현 하지 않고 단건 SELECT문을 실행 할 수 있는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "/fwt/updateStoreDetail.do")  **public** **void** updateStoreDetail(Map<String, Object> commandMap) **throws** Exception {  Map<String, String> selecthydId = **null**;  commandMap.put("hydWater", commandMap.get("HYD\_WTR\_NO"));  commandMap.put("srch\_ward", commandMap.get("WARD\_ID"));  selecthydId = winiService.selectByPk("fwtStoreManager.selectRecenthydId", commandMap);  HYD\_ID = String.*valueOf*(selecthydId.get("hydId"));  } |
| 매개변수 | - queryId : 단건 조회 처리 SQL mapping 쿼리 ID  - param : 단건 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map)  - return : SQL mapping 파일에서 지정한 resultClass/resultMap 에 의한 단일 결과 객체(보통 VO 또는 Map) |
| 반환값 | SQL mapping 파일에서 지정한 resultClass/resultMap 에 의한 단일 결과 객체(보통 VO 또는 Map) |

* + 1. list (jqgrid 사용시)

|  |  |
| --- | --- |
| 원형 | jqgridJason list(String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 jqgrid용 DATA Object로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value="board/selectAnswerList.do")  **public** @ResponseBody jqgridJason selectAnswerList(@ModelAttribute("searchVO") BoardVO boardVO,  @RequestParam Map<String, Object> commandMap) **throws** Exception {  jqgridJason resultData = **null**;  resultData = winiService.list("boardDAO.selectAnswerList", commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : 단건 조회 처리 SQL mapping 쿼리 ID  - param : 단건 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** JqgridJason  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과를 jqgrid 그리기 위한 JqgridJason 객체 담아 반환 |

* + 1. list (jqgrid 사용시 paging포함)

|  |  |
| --- | --- |
| 원형 | jqgridJason list(String queryId, String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 jqgrid용 DATA Object로 반환 하는 함수, 이 때 Object에 paginig이 셋팅 된다.  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value="board/selectAnswerList.do")  **public** @ResponseBody jqgridJason selectAnswerList(@ModelAttribute("searchVO") BoardVO boardVO,  @RequestParam Map<String, Object> commandMap) **throws** Exception {  jqgridJason resultData = **null**;  resultData = winiService.list("boardDAO.selectAnswerList", "boardDAO.selectAnswerListCnt", commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - queryId2 : LIST의 총 COUNT 계산을 위한 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** JqgridJason  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과(paging을 위한 count포함) 를 jqgrid 그리기 위한 JqgridJason 객체 담아 반환 |

* + 1. list\_json (winigrid 사용 시)

|  |  |
| --- | --- |
| 원형 | JSONObject list\_json(String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 winigrid용 DATA Object로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSampleList.do")  **public** @ResponseBody JSONObject selectSampleList(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  JSONObject resultData = **new** JSONObject();  resultData = winiService.list\_json("SampleDAO.selectSampleList", commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** JSONObject  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과를 winigrid 그리기 위한 JSONObject 객체 담아 반환 |

* + 1. list\_json (winigrid 사용 시 panging 포함)

|  |  |
| --- | --- |
| 원형 | JSONObject list\_json(String queryId, String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 winigrid용 DATA Object로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSampleList.do")  **public** @ResponseBody JSONObject selectSampleList(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  JSONObject resultData = **new** JSONObject();  resultData = winiService.list\_json("SampleDAO.selectSampleList", "SampleDAO.selectSampleListCnt", commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - queryId2 : LIST의 총 COUNT 계산을 위한 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** JSONObject  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과(paging을 위한 count포함)를 winigrid 그리기 위한 JSONObject 객체 담아 반환 |

* + 1. list\_easyui (easyuigrid 사용 시)

|  |  |
| --- | --- |
| 원형 | easyuigridJason list\_easyui(String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 easyuigrid용 DATA Object로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  **import** easyuiGridJasonDAO, eaysuiGridJasn 파일  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody easyuigridJason selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  easyuigridJason resultData = **new** easyuigridJason();  resultData = winiService.list\_easyui("Sample5DAO.selectSample5List",commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** easyuigridJason  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과를 easyuigrid 그리기 위한 easyuigridJason 객체 담아 반환 |

* + 1. list\_easyui (easyuigrid 사용 시 panging 포함)

|  |  |
| --- | --- |
| 원형 | easyuigridJason list\_easyui(String queryId, String queryId2, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 easyuigrid용 DATA Object로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  import easyuiGridJasonDAO, eaysuiGridJasn 파일  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody easyuigridJason selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  easyuigridJason resultData = **new** easyuigridJason();  resultData = winiService.list\_easyui("Sample5DAO.selectSample5List",  "Sample5DAO.selectSample5ListCnt", commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - queryId2 : LIST의 총 COUNT 계산을 위한 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** easyuigridJason  **:** SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과(paging을 위한 count포함)를 easyuigrid 그리기 위한 easyuigridJason 객체 담아 반환 |

* + 1. mult\_save (일괄 Service 처리)

|  |  |
| --- | --- |
| 원형 | easyuigridJason list\_easyui(String queryId, String queryId2, Map<String, Object> param) **throws** Exception; |
| 설명 | 한 개 이상의 insert, delete, update 문을 한 번에 실행하기 위한 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping("sample/sampleIUD.do")  **public** String sampleIDU(@ModelAttribute("searchVO") pageindexVO pageIndexVO, Map<String, Object> commandMap, ModelMap model) **throws** Exception {  jqgridJason resultData = **new** jqgridJason();  List<String> queryId = **new** ArrayList<String>(); // QueryID List  List<String> queryIUD = **new** ArrayList<String>(); // IUD List -> I: 등록, U:수정, D: 삭제  List<Map> save\_data = **new** ArrayList<Map>(); // 파라미터 데이터  Map<String, Object> map1 = commandMap;  queryId.add("WebLogDAO.logInsertWebLogTest");  queryIUD.add("I");  save\_data.add(map1);  queryId.add("WebLogDAO.logUpdateWebLogTest");  queryIUD.add("U");  save\_data.add(map1);  queryId.add("WebLogDAO.logDeleteWebLogTest");  queryIUD.add("D");  save\_data.add(map1);  resultData.setRows(save\_data);  **int** result = winiService.mult\_save(queryId, queryIUD, resultData);  **return** “result”  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID의 List  - queryIUD : 실행할 쿼리의 종류(INSERT,UPDATE,DELETE)의 List  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map)  - return : SQL mapping 파일에서 지정한 resultClass/resultMap 의 결과를 easyuigrid 그리기 위한 easyuigridJason 객체 담아 반환 |
| 반환값 | **-** String  **:** ‘1’ 수행완료  ’0’ 수행된 건이 없음 |

* + 1. list\_map (리스트 조회값을 Map에 담아 반환)

|  |  |
| --- | --- |
| 원형 | **public** Map<String, Object> list\_map(String queryId, String queryId2, Map<String, Object> param)  **throws** Exception |
| 설명 | 조회된 SQL SELECT 결과값과 list count를 Map으로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody Map<String,Object> selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  Map<string,Object> resultData = **new** HashMap<>();  resultData = winiService.list\_map("Sample5DAO.selectSample5List", "Sample5DAO.selectSample5ListCnt",commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - queryId2 : LIST의 총 COUNT 계산을 위한 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | **-** Map **:** SELECT LIST 조회된 값 |

* + 1. list\_map (List 형 반환 시 사용)

|  |  |
| --- | --- |
| 원형 | List list\_map(String queryId, Map<String, Object> param) **throws** Exception; |
| 설명 | 조회된 SQL SELECT 결과값을 List 형으로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody easyuigridJason selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  List resultData;  resultData = winiService.detail\_map("Sample5DAO.selectSample5List",commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | - List : 조회된 결과값 |

* + 1. detail\_map (List 반환 값을 Map에 담아 사용(count X))

|  |  |
| --- | --- |
| 원형 | **public** Map<String, Object> detail\_map(String queryId, Map<String, Object> param) **throws** Exception |
| 설명 | 조회된 SQL SELECT 결과값을 Map<String, Object> 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody Map<String, Object> selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  Map<String, Object> resultData = **new** HashMap();  resultData = winiService.detail\_map("Sample5DAO.selectSample5List",commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | - Map : SELECT LIST 조회된 값 |

* + 1. list\_map\_json (list를 json형태로 반환 시 사용)

|  |  |
| --- | --- |
| 원형 | **public** String list\_map\_json(String queryId, Map<String, Object> param) **throws** Exception |
| 설명 | 조회된 SQL SELECT 결과값을 JSON 형태로 반환 하는 함수  1. Controller.java 에 import  **import** winitech.com.service.WiniService;  2. Controller 내부에 선언  @Controller  **public** **class** FwtWriteController {  @Resource(name = "defaultService")  **private** WiniService winiService;  }  3. 실행하고자 하는 함수 내부에서 선언 실행  @RequestMapping(value = "sample/selectSample5List.do")  **public** @ResponseBody String selectSample5List(Map<String, Object> commandMap,  ModelMap model) **throws** Exception {  String resultData = **“”**;  resultData = winiService.list\_map\_json("Sample5DAO.selectSample5List",commandMap);  **return** resultData;  } |
| 매개변수 | - queryId : LIST 조회 처리 SQL mapping 쿼리 ID  - param : LIST 조회 처리 SQL mapping 입력 데이터(key)를 세팅한 파라메터 객체(보통 VO 또는 Map) |
| 반환값 | - String:  Map<List> 형태를 String형으로 감싸 JSON 으로 변환 할 수 있는 값을 반환 |

1. WINI Compoment (Js)
   1. wini\_cal.js

|  |  |
| --- | --- |
| 위치 | winiWeb/src/main/webapp/home/script/common/wini\_cal.js |
| 설명 | 달력과 날짜를 선택하는 UI를 제공하기 위한 공통컴퍼넌트 |
| 필수 선언 | <script type=*"text/javascript"* src=*"*<c:url value=*'/home/script/common/wini\_cal.js'*/>*"*></script> |

|  |  |
| --- | --- |
| 함수 | 설명 |
| wini\_com\_set\_detepick | INPUT BOX달력 UI 셋팅 |
| wini\_com\_set\_datepickstartend | 시작일, 종료일 달력 |
| wini\_cal\_setvalue | INPUT BOX 날짜 셋팅 |

* + 1. wini\_com\_set\_detepick (INPUT BOX달력 UI 셋팅)

|  |  |
| --- | --- |
| 원형 | **function** wini\_com\_set\_datepick(txtNm, defaultDayType, chk) |
| 설명 | 기간입력시 inputbox에 달력 UI 셋팅을 위한 함수  1. input box 생성  <input type=*"text"* id=*"calTest"* name=*"calTest"* maxlength=*"20"* value=*""* />  2. script 함수 선언  <script type=*"text/javascript"*>  $(**function**(){  wini\_com\_set\_datepick("calTest",“0”,**true**); // 달력 셋팅 함수  }  </script>  적용전)    적용후) |
| 매개변수 | - textNm : inputBox ID 값  - defaultDayType : 셋팅할 기본날짜 코드 (wini\_cal\_setValue 함수참고)  - chk : 날짜형식체크 |
| 반환값 | 없음 |

* + 1. wini\_com\_set\_datepickStartEnd (시작일, 종료일 달력)

|  |  |
| --- | --- |
| 원형 | **function** wini\_com\_set\_datepickStartEnd(txtNmStart, txtNmEnd, defaultDayStart, defaultDayEnd, chk) |
| 설명 | 2개의 INPUT BOX에 시작일을 기준으로 종료일까지를 셋팅 할 수 있는 달력 UI  1. 시작일, 종료일 inputBox 선언.  <input type=*"text"* id=*"calTeststart"* name=*"calTeststart"* value=*""*></input>  <input type=*"text"* id=*"calTestEnd"* name=*"calTestEnd"* value=*""*></input>  2. 스크립트 선언  <script type=*"text/javascript"*>  $(**function**(){  // 달력 셋팅 함수  wini\_com\_set\_datepickStartEnd("calTeststart","calTestEnd",'','',**true**);  }  </script>  적용전)    적용후) |
| 매개변수 | - textNmStart: 시작일 inputBox ID 값  - textNmEnd: 종료일 inputBox ID 값  - defaultDayType : 셋팅할 기본날짜 코드 \*wini\_cal\_setValue 함수참고  - chk: 날짜형식체 |
| 반환값 | 없음 |

* + 1. wini\_cal\_setValue (INPUT BOX 날짜 셋팅)

|  |  |
| --- | --- |
| 원형 | **function** wini\_cal\_setValue(value, dayvalue) |
| 설명 | DATE 객체를 활용하여 오늘날짜, 하루전, 한달전, 이번달 1일, 저번달 1일, 오늘날짜기준으로 +,- 된 날짜를 셋팅 할 수 있는 함수  1. input box 생성  <input type=*"text"* id=*"* calTeststart *"* name=*"* calTeststart *"* maxlength=*"20"* value=*""* />  2. script 함수 선언  <script type=*"text/javascript"*>  $(**function**(){  $("#calTeststart").val(wini\_cal\_setValue("0")); 당일 날짜 셋팅  $("#calTeststart").val(wini\_cal\_setValue("3")); 당월 1일 셋팅  $("#calTeststart").val(wini\_cal\_setValue("9","10")); 당일날짜 +10일  }  </script>  적용후)  당일날짜  당월 1일  당일날짜 +10일 |
| 매개변수 | - value : 셋팅할 날짜 코드  - dayvalue: 셋팅할날짜 + - 일수 |
| 반환값 | 없음 |

* 1. wini\_code.js

|  |  |
| --- | --- |
| 위치 | winiWeb/src/main/webapp/home/script/common/wini\_code.js |
| 설명 | SELECT LIST에 코드 셋팅 및 콤보 단계로 나누어 셋팅 하기 위한 함수파일 |
| 필수 선언 | <script type=*"text/javascript"* src=*"*<c:url value=*'/home/script/common/wini\_code.js'*/>*"*></script> |

|  |  |
| --- | --- |
| wini\_code 함수 목록 | |
| CodeUtil.setCodeList | 코드셋팅(콤보 포함) |
| CodeUtil.makeRadio | 라디오버튼 셋팅 |

* + 1. CodeUtil.setCodeList (코드그룹을 이용한 최대 2단계 SELECT BOX구현)

|  |  |
| --- | --- |
| 원형 | CodeUtil.setCodeList = **function**(objType, makeObj, codeLevel, upCd1, upCd2, firstItem, selectItem) |
| 설명 | 코드그룹별로 조회된 코드값을 SELECT BOX 리스트에 자동으로 셋팅을 할 수 있다. (최대 2단계)   1. 코드 inputBox 선언.   <select id=*"cmbCode"* name=*"cmbCode"* style="width: *150px*"/>  <select id=*"cmbCodeSub"* name=*"cmbCodeSub"* style="width: *150px*"/>  2. script 함수 선언  <script type=*"text/javascript"*>  $(**function**(){  CodeUtil.setCodeList('combo', 'cmbCode', '2', '001', '', '선택', '');  $('#cmbCode').on('change', **function**() {  CodeUtil.setCodeList('combo', 'cmbCodeSub', '2', '001', **this**.value, '선택', '');  }  });  </script>  적용후) |
| 매개변수 | - objType : 생성할 컨트롤의 종류 ('combo', 'radio')  - makeObj : 생성할 object ID  - codeLevel : 코드 조회단계  - upCd1 : 1단계 코드값  - upCd2 : 2단계 코드값  - firstItem : combo에 보여질 첫번째 값(선택, 전체, 등), null 로 보내면 첫번째 값이 없이 생성  - selectItem : 생성 후 초기 선택되어지는 값 |
| 반환값 | 없음 |

* + 1. CodeUtil.setCodeList (라디오버튼 셋팅)

|  |  |
| --- | --- |
| 원형 | CodeUtil.setCodeList = **function**(objType, makeObj, codeLevel, upCd1, upCd2, firstItem, selectItem) |
| 설명 | 3.2.1의 함수의 추가기능으로 objType의 매개변수에 combo 대신 radio를 매개수로 입력하면 된다  1.Div 셋팅  <div id=*"testCodeObj"*></div>  2.스크립트 선언  <script type=*"text/javascript"*>  $(**function**(){  CodeUtil.setCodeList('radio', 'testCodeObj', '2', '001', '', **null**, '');  });  </script>  출력 화면)  EMB0000535c14df |
| 매개변수 | - objType : 생성할 컨트롤의 종류 ('combo', 'radio')  - makeObj : 생성할 object ID  - codeLevel : 코드 조회단계(2, 3)  - upCd1 : 1단계 코드값  - upCd2 : 2단계 코드값  - firstItem : combo에 보여질 첫번째 값(선택, 전체, 등), null 로 보내면 첫번째 값이 없이 생성  - selectItem : 생성 후 초기 선택되어지는 값 |
| 반환값 | 없음 |

* 1. wini\_default.js

|  |  |
| --- | --- |
| 위치 | winiWeb/src/main/webapp/home/script/common/wini\_default.js |
| 설명 | AJAX 처리, 유효성 체크 등 실무에서 많이 사용 되는 공통컴퍼넌트 |
| 필수 선언 | <script type=*"text/javascript"* src=*"*<c:url value=*'/home/script/common/wini\_default.js'*/>*"*></script> |

|  |  |
| --- | --- |
| 함수 | 설명 |
| wini\_getAjax | Ajax 실행 |
| wini\_submit\_ingView | 진행중 표시 |
| MM\_moveURL | 강제 화면 이동 |
| zoom\_it | 화면 확대/축소 |
| checkBrowser | 브라우저 버전 |
| gfn\_check\_tel | 전화번호 유효성 체크 |
| gfn\_check\_tel2 | 전화번호 유효성 체크2 |
| gfn\_CheckValue | 영문 & 숫자 입력 체크 |
| emailcheck | EMAIL 유효성체크 |
| gfn\_checkMobileScreen | 스마트폰 유무 체크 |
| gfn\_isNum | 숫자만 체크 |
| password\_check | 비밀번호 유효성 체크 |
| gfn\_check\_number | 정수, 실수 입력 |
| gfn\_extChk | 파일 확장자 체크 |
| gfn\_openPop | 진행중 표시 |
| wini\_myDateChk | 날짜 유효성 체크 |

* + 1. wini\_getAjax

|  |  |
| --- | --- |
| 원형 | function wini\_getAjax(ajax\_set) |
| 설명 | AJAX를 실행하기 위한 함수  1. Div 셋팅  <div id=*"testObj"*></div>  2. 스크립트 선언  <script type=*"text/javascript"*>  $(function(){  var ajax\_set =  {  url:"../sample/test.do?",  param:"TEST=" +"TEST",  return\_fn:function(jdata){$("#testObj").html("통신성공"); }  }  wini\_getAjax(ajax\_set);  }  });  </script>  -Controller  @RequestMapping(value = "sample/test.do")  public @ResponseBody String selectSampleList(Map<String, Object> commandMap,  ModelMap model, HttpServletResponse response) throws Exception {  String resultData="";    if(String.*valueOf*(commandMap.get("TEST")).equals("TEST"))  resultData = "TEST";    return resultData;  } |
| 매개변수 | - ajax\_set : AJAX옵션을 설정한 변수 |
| 반환값 | 없음 |

* + 1. wini\_submit\_ingView (진행중 표시)

|  |  |
| --- | --- |
| 원형 | function wini\_submit\_ingView(flag) |
| 설명 | 저장, 처리 등 이벤트 발생시 진행 중 표시(wini\_gird.css 필요)  1.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  var d=$("<div id=\"wini\_submit\_ing\"></div>").appendTo("body");  d.hide();  var d2=$("<div id=\"wini\_window\_mask\_cont\"><img src=\""+"/"+wini\_getContextName()+"/home/images/wini\_grid/buffering.gif\"/><br/>처리중입니다.</div>").appendTo("body");  d2.hide();  }  wini\_submit\_ingView(true);  </script>  -Controller  @RequestMapping(value = "sample/test.do")  public @ResponseBody String selectSampleList(Map<String, Object> commandMap,  ModelMap model, HttpServletResponse response) throws Exception {  String resultData="";    if(String.*valueOf*(commandMap.get("TEST")).equals("TEST"))  resultData = "TEST";    return resultData;  }  EMB0000535c14e3적용후) |
| 매개변수 | - flag : 조회 중 화면 on/off에 대한 참,거짓 값으로 ture or false 입력 |
| 반환값 | 없음 |

* + 1. MM\_moveURL (강제 화면 이동)

|  |  |
| --- | --- |
| 원형 | function MM\_moveURL(url) |
| 설명 | 이동 하고자하는 URL로 이동  1. 이동하고자하는 URL 입력  MM\_moveURL(“./test,do”); |
| 매개변수 | - url : 이동하고자하는 화면의 URL |
| 반환값 | 없음 |

* + 1. zoom\_it (화면 확대/축소)

|  |  |
| --- | --- |
| 원형 | Function zoom\_it(n) |
| 설명 | 화면 크기 단계별 확대/축소  1.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  zoom\_it(2);  });  </script>  적용전)    적용후) |
| 매개변수 | - n : 확대/축소 단계 |
| 반환값 | 없음 |

* + 1. checkBrowser (브라우저 버전)

|  |  |
| --- | --- |
| 원형 | function checkBrowser() |
| 설명 | 현재 브라우저의 브라우저 종류와 버전을 확인 할 수 있다.  1.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  console.log(checkBrowser());//크롬에서만 사용가능  });  </script>  적용후)  EMB0000535c134b |
| 매개변수 | 없음 |
| 반환값 | name: 현재 브라우저의 종류를 반환한다  version: 현재 브라우저의 버전을 반환한다 |

* + 1. gfn\_check\_tel (전화번호 유효성 체크)

|  |  |
| --- | --- |
| 원형 | function gfn\_check\_tel(input) |
| 설명 | INPUT BOX로 입력 받은 전화번호에 대하여 유효성 체크한다  1. input Box 선언  <input type=*"text"* id=*"telCheck"* name=*"telCheck"* value=*""*></input>  2.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  $("#telCheck").blur(function(){ gfn\_check\_tel(this);  });  </script>  적용후)  EMB0000535c136b |
| 매개변수 | - input : 유효성 체크 받을 전화번호값 |
| 반환값 | Boolean 값  유효:true, 비유효:false |

* + 1. gfn\_check\_tel2 (전화번호 유효성 체크2)

|  |  |
| --- | --- |
| 원형 | function gfn\_check\_tel2(id, tel) |
| 설명 | 전화번호 유효성 체크한다는 부분에서 3.3.6과 같지만 inputbox 3개 두는것에서 차이가 있다.  1. input Box 선언  <input type=*"text"* id=*"telCheck01"* name=*"telCheck"* value=*""*></input>  <input type=*"text"* id=*"telCheck02"* name=*"telCheck"* value=*""*></input>  <input type=*"text"* id=*"telCheck03"* name=*"telCheck"* value=*""*></input>  <button id=*"telCheckButton"*>전화번호체크</button>  2.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  $("#telCheckButton").click(function(){gfn\_check\_tel2("telCheck", "tel");  });  </script>  EMB0000535c13a2적용후)  \*유효하지 않은 전화번호 체크시 |
| 매개변수 | - id: 체크할 input Box의 id  - tel: 체크 받을 번호의 종류 (전화번호:tel, 팩스:fax, 회사전화:co 휴대폰:cell) |
| 반환값 | - true : 유효한 전화번호  - false: 유효하지 않은 전화번호 |

* + 1. gfn\_CheckValue (영문 & 숫자 입력 체크)

|  |  |
| --- | --- |
| 원형 | function gfn\_CheckValue(id, name) |
| 설명 | 영문 & 숫자 입력 체크  1.input Box 선언  <input type=“text” id=“loginId”/>  2.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  gfn\_CheckValue(‘loginId’, ‘로그인ID’);  });  </script> |
| 매개변수 | - id : 체크 할 입력 ID  - name : ID의 NAME ex)‘로그인 ID’ |
| 반환값 | - true : 영문 & 숫자만 입력  - false : 영문 & 숫자 외 입력 |

* + 1. emailCheck (email 유효성체크)

|  |  |
| --- | --- |
| 원형 | function emailCheck(email) |
| 설명 | email 유효성체크  1.input Box 선언  <input type=*"text"* id=*"emailCheck"* name=*"emailCheck"* value=*""*></input>  2.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  $("#emailCheck").blur(function(){ emailCheck(this);  });  </script> |
| 매개변수 | - email : 입력받을 ID 값 |
| 반환값 | 없음 |

* + 1. gfn\_checkMobileScreen (스마트폰 유무 체크)

|  |  |
| --- | --- |
| 원형 | function gfn\_chkMobileScreen() |
| 설명 | 현재 접속한 기기의 해상도를 체크하여 모바일화면, 데스크톱을 구별 한다  1.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  If(gfn\_chkMobileScreen()){  alert(“스마트폰입니다”);  }  });  </script> |
| 매개변수 | 없음 |
| 반환값 | 없음 |

* + 1. gfn\_isNum (숫자만 체크)

|  |  |
| --- | --- |
| 원형 | function gfn\_isNum(input) |
| 설명 | 입력한 값이 숫자로만 입력 되었는지 체크한다.  1.INPUT BOX 선언  <input type=*"text"* id=*"isNumtest"* name=*"isNumtest"* value=*""*></input>  2.스크립트 선언  <script type=*"text/javascript"*>  Var input = $(“#inNumtest”).val()  if(gfn\_isNum(input)){  alert(“숫자만체크”)  }  </script> |
| 매개변수 | - input : 숫자 체크를 위한 입력 값 |
| 반환값 | 없음 |

* + 1. password\_check (비밀번호 유효성 체크)

|  |  |
| --- | --- |
| 원형 | function password\_check() |
| 설명 | 비밀번호 유효성 체크 |
| 사용법 | 1.input box 선언  input type=*"text"* id=*"id\_check"* name=*"id\_check"* value=*""* ></input>  <input type=*"password"* id=*"user\_pw"* name=*"user\_pw"* value=*""* autocomplete=*"off"*></input>  <input type=*"password"* id=*"user\_pw\_identify"* name=*"user\_pw\_identify"* value=*""*autocomplete=*"off"*></input>  <button id=*"passwdCheckButton"*>비밀번호체크</button>  2.스크립트 선언  <script type=*"text/javascript"*>  $(function(){  $("#passwdCheckButton").click(function(){password\_check();  });  </script> |
| 매개변수 | - input: 비밀번호 유효성을 체크하기위한 값 |
| 반환값 | - true: 유효한 값  - false: 유효하지않은 값 |

* + 1. gfn\_check\_number (값 입력시 정수, 실수 제한)

|  |  |
| --- | --- |
| 원형 | function gfn\_check\_number(obj, type, tlen, flen, comma) |
| 설명 | 정수 및 실수 type 정하여 범위에 따라 입력값 제한  1. input box, keyup옵션 선언  <input type=*""* onkeyup="gfn\_check\_number(this,'int',5,2,'Y');" /> |
| 매개변수 | - obj : this (고정)  - type : int => 정수(소수 입력 안 됨) / -int => -정수(소수 입력 안 됨) / float => 실수(소수 입력 가능) / -float => -실수 (소수 입력 가능)  - tlen : tlen => 총 자리수 (정수 입력 시 tlen에서 flen 뺀 결과가 정수자리수)  - flen : 소수 자리수 (~까지)  - comma : Y => 세자리마다 콤마(,) 자동입력 N => 콤마 미적용 |
| 반환값 | 없음 |

* + 1. gfn\_extChk (파일 확장자 체크)

|  |  |
| --- | --- |
| 원형 | function gfn\_extChk(fileType, fileObj) |
| 설명 | 파일 확장자를 체크하기 위한 함수  1.input Box 선언  <input id=*"fileupload"* type=*"file"* name=*"files[]"* multiple=*"multiple"*>  2. SCRIPT 선언  <script type=*"text/javascript"*>  Function fileExtCheck(){  var flag = gfn\_extChk(“DOC”, “fileupload”);  if(flag){  alert(“업로드가능한 확장자입니다”)  }  }  </script> |
| 매개변수 | - fileType : 기준이 되는 type (DOC,IMG,MOV,doc,docx)  - fileObj : 파일 input 객체 |
| 반환값 | - true : 첨부 가능한 확장자  - false : 첨부 불가능한 확장자 |

* + 1. gfn\_openPop (진행중 표시)

|  |  |
| --- | --- |
| 원형 | function gfn\_openPop(option) |
| 설명 | 팝업창을 띄우기 위한 함수  1.팝업호출 버튼 선언  <button id=*"popupButton"* onclick="open()">팝업 호출</button>  2.스크립트 선언  <script type=*"text/javascript"*>  Function open(){  gfn\_openPop(  {  url : "../sys/test.do",  name : "test",  width : 600,  height : 200,  scrollbars : "yes",  resizable : "no",  param:{},  callback : ""  });  }  wini\_submit\_ingView(true);  </script>  -Controller  @RequestMapping("sys/test.do ")  public String letterFileUploadPopup(Map<String, Object> commandMap, ModelMap model) throws Exception {  return "sys/test";  } |
| 매개변수 | - option : 사용법에서 보여지는 각 기능 입력 |
| 반환값 | 없음 |

* + 1. wini\_myDateChk (입력된 날짜 유효성 체크)

|  |  |
| --- | --- |
| 원형 | function wini\_myDateChk(obj, type) |
| 설명 | INPUT BOX를 통해 입력된 날짜의 유효성(윤년 포함)을 체크한다  1. INPUT BOX 선언  <input type=*"text"* id=*"*inputBox*"* name=*"*inputBox*"* value=*""*></input>  2. 스크립트 선언  <script type=*"text/javascript"*>  $(function(){  $("#inputBox“).blur(function(){ wini\_myDateChk(this, 'yyyymmdd');  }  </script> |
| 매개변수 | - obj : 체크받을 객체의 ID값  - type :기준되는 날짜형식(‘yyyymmdd’,‘yyyymm’,‘yyyy’,‘mm’) |
| 반환값 | 없음 |

* 1. wini\_gird.js

|  |  |
| --- | --- |
| 위치 | winiWeb/src/main/webapp/home/script/common/wini\_grid.js |
| 설명 | 많이 사용 되는 grid 라이브러리를 함수 수정 가능한 상태로 정리 해놓은 파일 |
| 필수 선언 | <script type=*"text/javascript"* src=*"*<c:url value=*'/home/script/common/wini\_grid.js'*/>*"*></script> |

* + 1. winigrid 그리는 법

|  |  |
| --- | --- |
| 원형 | - |
| 설명 | 기본적인 winigrid를 그리기 위한 과정  - HTML  ◉검색조건을 위한 form  <form name=*"frmS"* id=*"frmS"* method=*"post"* enctype=*"multipart/form-data"* onSubmit="Search(); return false;">  <input type=*"text"* id=*"srch\_name"* name=*"srch\_name"* value=*"“* class=*"i\_text"* style="width:*98%*; ime-mode:*active*;"/>  <select id=*"srch\_use\_yn"* name=*"srch\_use\_yn"* style="width: *90px*">  <option value=*""* selected="selected">전체</option>  <option value=*"Y"* selected="selected">사용</option>  <option value=*"N">* 사용안함</option>  </select>  </form>  ◉grid를 그릴 테이블  <table id=*"grid\_dataList"*></table>  <div id=*"grid\_dataList\_page"*></div>  - Jqeury  ① gird 내용 셋팅  var tb1\_wini\_grid =  {  url:"./selectSampleList.do", 데이터를 조회하기위한 url  mtype : 'POST',  postData:{},  colNames:[  {"head" : ['ID', '이름', '내용', '사용여부', '작성자']} 헤더의 이름  ],  colModel:[  {name:'id', index:'id', width:120, align:"center", hidden:false, coltype:""},  {name:'name', index:'name', width:150, align:"center", hidden:false, coltype:""},  {name:'description', index:'description', width:0, align:"left", hidden:false, coltype:""},  {name:'useYn', index:'useYn', width:80, align:"center", hidden:false, coltype:""},  {name:'regUser', index:'regUser', width:120, align:"center", hidden:false, coltype:""}  ],  autowidth: true,  rowNum: '10', 그리드의 줄 수  rownumbers: true,  viewrecords: true,  viewpager: true,  rowCntView: true,  multiselect: false, //첫 컬럼에 체크박스 생성  multiselectPosition:"left", // 체크박스 위치 left or rigit  headmerge: false,  rowscroll: false,  dataheight:285,  //datamerge: [2, 3],  onClickRow: "grid\_Detail\_Info",  tableid: "grid\_dataList",  page:1,  pagerurl: "listGird",  pager: "grid\_dataList\_page"  }  ② 그리드 호출  $(function(){  listGird(1);  });  ③ 그리드호출을 위한 함수  function listGird(pagenum){  tb1\_wini\_grid.postData = {  srch\_name:$('#srch\_name').val(),  srch\_use\_yn:$('#srch\_use\_yn').val()  }; //form 선언이아닌 ajx 방식  tb1\_wini\_grid.formName = "#frmS"; //HTML에서 선언한 form Name  tb1\_wini\_grid.dummyList = false;  tb1\_wini\_grid.page = pagenum; // 첫셋팅할 페이지. 기본: 1  winigrid\_call(tb1\_wini\_grid); 그리드호출 위한 함수  }  - Controller  @RequestMapping(value = "sample/selectSampleList.do")  public @ResponseBody JSONObject selectSampleList(Map<String, Object> commandMap,ModelMap model) throws Exception {  JSONObject resultData = new JSONObject();  try {  ①winiService(winiService.Java)의 선언이 필요하다.  resultData = winiService.list\_json("SampleDAO.selectSampleList", "SampleDAO.selectSampleListCnt", commandMap);  } catch (Exception e) {  winigridJasonDAO winigrid = new winigridJasonDAO();  resultData = winigrid.makeErrMsg("-1", e.toString()); // 에러 메시지 생성  log.error(e.toString());  }  return resultData; sql문을 통해 data를 조회한후 JSONObject로 내려보내면 data값이 그리드로 셋팅  }  적용후) |
| 매개변수 | 없음 |
| 반환값 | 없음 |

* + 1. grid\_detail\_Info (그리드 목록 클릭)

|  |  |
| --- | --- |
| 원형 | function grid\_Detail\_Info(nRow) |
| 설명 | 그리드 목록 클릭 & 그리드 로우 객체 가져오기  1. 그리드 그리기  - 3.4.1 wini그리드 그리는법 참조.  2. 그리드 가로줄(row) 클릭  \* 그리드 선언시 옵션 onClickRow: "grid\_Detail\_Info", 추가 필수  -SCRIPT 선언  function grid\_Detail\_Info(nRow){  var data = "";  var id = "";  //그리드 로우 객체  data = winigrid\_getRowObj("grid\_dataList", nRow);  //- nRow : nRow번째 줄의 객체 가져오기  id = winigrid\_getRowValue(data, "id");  } |
| 매개변수 | - nRow : 클릭한 그리드의 n 번째 줄 |
| 반환값 | 없음 |

1. 변경 내용: 변경이 발생되는 위치와 변경 내용을 자세히 기록(장/절과 변경 내용을 기술한다.) [↑](#footnote-ref-1)