

**TMT-COMMON**

소스코드 NamingRule

V1.0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 구분 | 성명 | 일자 | 의 견 | 서명 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Document #:

**문서 이력**

| 버전 | 작성일 | 변경내용[[1]](#footnote-1) | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 1.0 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[1. 개요 1](#_Toc534358509)

[1.1 목적 1](#_Toc534358510)

[1.2 적용 범위 1](#_Toc534358511)

[2. JAVA 소스 1](#_Toc534358512)

[2.1 공통 1](#_Toc534358513)

[2.2 속성(Attribute)명 1](#_Toc534358514)

[2.3 메소드(Method), 클래스(Class) 명 1](#_Toc534358515)

[2.4 SQL(Query) ID, 파일명 2](#_Toc534358516)

[2.5 요청 매핑(Request Mapping) 3](#_Toc534358517)

[2.6 객체 4](#_Toc534358518)

[2.7 변수(Variable) 4](#_Toc534358519)

[2.8 상수(Constant) 5](#_Toc534358520)

[2.9 패키지(Package) 구조 5](#_Toc534358521)

[3. 클라이언트(UI) 소스 6](#_Toc534358522)

[3.1 표시기준 6](#_Toc534358523)

[3.2 파일(File)명 6](#_Toc534358524)

[3.3 eXbuilder6(clx) 소스 8](#_Toc534358530)

[3.4 JavaScript 소스 11](#_Toc534358532)

1. 개요
   1. 목적

본 문서는 소스코드 개발시 일관된 명명 규칙을 정의함으로써 전체 개발 소스에 대한 일관성을 유지함에 목적이 있다.

* 1. 적용 범위

프로젝트 수행 시 작성되는 소스코드에 대하여 적용한다. 단, 프로젝트 관리 목적으로 사용되는 일부 소스코드는 예외로 한다.

1. JAVA 소스
   1. 공통

* 복합된 단어의 경우 각 단어의 첫 글자는 대문자, 나머지는 소문자로 한다. 단, 첫 단어의 첫 글자는 소문자로 한다.  
  (예) isValidation, getQueryForList
* Underscore(\_)를 사용하지 않는다.
* 예외 사항 : JAVA 클래스(Class)명은 첫 단어의 첫 글자를 **대문자**로 명명한다.  
  (예) CmnCodeController, CmnCodeService, CmnCodeServiceImpl
  1. 속성(Attribute)명
* Table의 column명과 매칭되는 경우 column명과 동일하게 사용한다.

(예) strUserNm(사용자명), strStudNo(학번)

* 1. 메소드(Method), 클래스(Class) 명
* 동사+명사의 구조로 명명한다.
* Controller, Service 클래스명은 업무별로 적절한 명칭을 사용한다.
* 하단 명명 시 [표 2-1] 명명 규칙 참조

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | service | Class | method |
| Controller | Controller | 프로그램ID + Controller  ex) CmnCodeController | onLoad  list  list+Xxx  update+Xxx  save  save+Xxx |
| Service | Service(인터페이스)  ServiceImpl(구현체) | 업무(3)+구분+ Service  ex) CmnCodeService  CmnCodeServiceImpl | select+업무(3)+Xxx(d)+  Object(단건 조회)  ex) selectCmnCodeNm |
| select+업무(3)+Xxx(d)(단건 조회)  ex) selectCmnCode |
| select+업무(3)+Xxx(d)+ List(다건 조회)  ex) selectCmnCodeList |
| insert+업무(3)+Xxx(d)(입력)  ex) insertCmnCode |
| update+업무(3)+Xxx(d) (수정)  ex) updateCmnCode |
| delete+업무(3)+Xxx(d) (삭제)  ex) deleteCmnCode |

[표 2-1 명명 규칙]

* 1. SQL(Query) ID, 파일명

데이터베이스에 대한 쿼리(Query)는 ORM 프레임워크인 Mybatis 문법을 준수하며, XML 파일에 기술하여 관리한다.

* 파일명은 기본적으로 중메뉴 단위로 분리한다.
* SQL 파일내의 작성한 쿼리가 5,000 라인을 넘어가는 경우 또는 개발자가 동일 업무파트를 같이 개발해야되는 경우에 한하여 파일을 분리하여 관리할 수 있다.

(cmn-base01.xml, cmn-base02.xml 등등)

* 업무에 공통적인 쿼리들만을 그룹화하여 별도 파일로 분리할 수 있다.
* 각 파일내의 SQL namespace 명은 해당 파일명과 동일하게 지정한다.
* SQL 아이디는 모든 쿼리 파일에서 중복되지 않는 명칭을 사용해야 하며, 해당 쿼리를 호출하는 서비스(Service) 클래스의 메소드명을 기본으로 한다.
* 자바 유틸에 의해 생성되는 원 테이블들에 대한 CUD 쿼리는 tables 접미어를 붙인 파일에 추가 관리한다.(cmn-tables01.xml, cmn-tables02.xml)

■ SQL 파일명

|  |
| --- |
| 명명규칙: 업무(3)+**구분**{3~7}+순번(2).xml  \*\* [구분]은 해당 쿼리의 중메뉴에 대한 의미있는 영문을 사용한다.  \*\* 업무파트별 공통적인 쿼리들을 별도 분리하여 하나의 쿼리 파일로 작성하는 경우에는 common이라는 영문을 사용한다. (분리가 필요한 경우에만 해당됨)  예시) cmn-base01.xml, cmn-usermgt01.xml, aps-common.xml |

■ SQL Mapper Namespace명

|  |
| --- |
| 명명규칙: 해당 SQL 파일명과 동일   * SQL 파일명이 cmn-base01.xml인 경우, 매퍼의 네임스페이스명 역시 cmn-base가 됨   예시) <mapper namespace="cmn-base01"> |

■ SQL ID명

|  |
| --- |
| 명명규칙: 서비스(Service) 클래스의 호출 메소드명   * CmnServiceImpl 클래스의 selectCmnCodeList() 메소드에서 호출하는 SQL인 경우에는 해당 SQL의 ID는 selectCmnCodeList이 된다.   예시) <select id=”selectCmnCodeList”> |

■ SQL 동적 파라메터 변수명

쿼리(SQL)문에 매핑되는 동적 변수들에 대한 명명 규칙을 정의한다.

|  |
| --- |
| 명명규칙:   * DB컬럼에 직접적으로 바인딩되는 변수는 해당 컬럼명과 동일하게 대문자 영문으로 변환하여 사용한다. * 그 외 플래그(Flag) 및 조건 분기등을 위한 변수는 파라메터 문자 그대로 사용한다.   예시) SELECT CD  , CD\_NM  FROM CMN0101  WHERE CD\_CLS = #{**CD\_CLS**}  <if test=”**strFlag** and **strFlag** eq ‘Y’.toString()”>  AND CD\_USG\_01 = ‘TEST’  </if> |

* 1. 요청 매핑(Request Mapping)

서버단 컨트롤러(Controller) 클래스는 사용자 화면으로부터의 요청을 받아서 처리하는 역할을 담당한다. 따라서, 각 컨트롤러 클래스들의 메소드에서는 어떤 요청을 처리를 해야 하는지를 명확히 지정해주어야 한다.

<CmnMessageController 클래스의 선언 예시>

|  |
| --- |
| /\*\*  \* 시 스 템 : 공통  \* 단위시스템 : 공통시스템  \* 프로그램명 : 메시지관리  \* 설 명 : 서버단 오류 및 업무 메시지를 관리한다.  \*  \* 이력사항  \* 2018. 8. 16. SUL 최초작성  \*/  @Controller  **@RequestMapping**("**/CmnMessage**")  **public** **class** CmnMessageController {  …  } |

* 컨트롤러 클래스 전체에 대한 요청을 매핑하는데 사용되며, 기본적으로 해당 클래스의 명칭을 사용한다. (단, suffix인 ‘Controller’ 단어는 제외한 명칭 사용)

|  |
| --- |
| 매핑 규칙: **/컨트롤명**  예) /CmnMessage, /CmnCode |

컨트롤러 클래스의 각 메소드별로 클라이언트의 요청에 대한 매핑 URL을 추가한다.

<CmnMessageController 클래스의 list 메소드 처리 예시>

|  |
| --- |
| **@RequestMapping("/list.do")**  **public** ModelAndView list(HttpServletRequest request, HttpServletResponse response, RequestData requestData) **throws** Exception {  //메시지 목록을 반환한다.  } |

해당 메소드 상위의 어노테이션인 @RequestMapping 선언을 통해, 명시된 URL 요청에 대해 해당 컨트롤러 클래스가 요청을 자동 처리하게 되는 것이다.

|  |
| --- |
| 매핑 규칙: **/메소드명 + ‘.do’**  예)/onLoad.do, /list.do, /save.do |

* 1. 객체
* Class 명과 반드시 동일하게 할 필요는 없으며, 최소한의 의미가 전달될 수 있는 명사로 명명한다.

예) class 명과 동일하게 사용  
 => CmnCodeService, cmnCodeService;

* 1. 변수(Variable)
* 변수의 첫 글자는 소문자로 시작한다.
* static 과 final 변수는 대문자를 사용한다. (ex- PROGRASS\_RATE)
* 변수가 지칭하는 데이터를 표현하는 영문을 찾을 수 없는 경우에는 한글표현에 해당하는 영어표현을 사용한다.

|  |
| --- |
| * ㄱ -> g, ㄴ -> n, ㄷ -> d, ㄹ(초성) -> r, ㄹ(종성) -> l, ㅁ -> m, ㅂ -> b, ㅅ -> s, * ㅇ -> ng, ㅈ -> j, ㅊ -> ch, ㅋ -> k, ㅌ -> t, ㅍ -> p, ㅎ -> h, * ㅏ -> a, ㅐ -> ae, ㅑ -> ya, ㅒ -> yae, ㅓ -> eo, ㅔ -> e, ㅕ -> yeo, ㅖ -> ye, ㅗ -> o, ㅘ -> oa, ㅚ -> oi, ㅛ -> yo, ㅜ -> u, ㅟ -> ui, ㅠ -> yu, ㅡ -> eu, ㅢ -> eui, ㅣ -> i |

* 해당 변수가 특정 DB 테이블의 컬럼의 값을 나타내는 경우, 해당 DB의 컬럼명에서 언더바(\_)를 제외하고 붙인 컬럼명을 지역변수명으로 사용한다.
* 변수의 데이터 타입에 따른 지역변수 명칭은 아래의 규칙을 권고한다.

|  |  |
| --- | --- |
| 데이터 타입 | 내용 |
| String | str + 명사로 선언한다.  ex) strEmpNo |
| Int | i + 명사로 선언한다.  ex) iCnt |
| Boolean | is + 명(동)사로 선언한다.  ex) isResult |
| List | 명사 + List로 선언한다.  ex) resultList |

* 1. 상수(Constant)
* 변수명 전체를 대문자로 하고 단어 연결에는 Underscore(\_)를 사용한다.  
  예) MINIMUM\_BALANCE
  1. 패키지(Package) 구조
* Java package 구조는 “<단위업무구분 코드>.<레이어>”로 구분한다.
* <레이어>은 controller, service/service.impl, sqlmap를 사용한다.

예) org.tmt.iis.cmn, org.tmt.iis.aps

|  |  |
| --- | --- |
| 파일 유형 | 패키지 위치 |
| 컨트롤러(Controller) 파일 | 컨트롤러 JAVA 클래스 파일은 [단위시스템]별 하위의 controller 패키지에 위치한다.  예) org.tmt.iis.cmn.controller |
| 서비스(Service) 파일 | 서비스 인터페이스 클래스 파일은 [단위시스템]별 하위의 service 패키지에 위치한다.  예) org.tmt.iis.cmn.service |
| 서비스 구현체(ServiceImpl) 파일 | 서비스 구현체(implementation) JAVA 파일은 해당 서비스 인터페이지가 위치한 패키지 하위의 impl 패키지에 위치한다.  예) org.tmt.iis.cmn.service.impl |
| ORM SQL-Mapper XML 파일 | 실행 쿼리문(SQL)을 작성한 Mapper XML 파일은 [단위시스템]별 하위의 sqlmap 패키지에 위치한다.  예) org.tmt.iis.cmn.sqlmap  \*용어) sqlmap: ORM 프레임워크인 Mybatis에서 사용하는 Sql Mapper 용어의 약칭임 |

1. 클라이언트(UI) 소스
   1. 표시기준

* { } : 선택
* [ ] : 자리수
* X…X : 제한없음
* XXX : 자리수 고정 예) XXXX 4자리, XXXXX 5자리
* 단위시스템 : 3자리 단위시스템명칭을 따르도록 한다.
* 화면유형 : 대문자로 기술한다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 유형 | 입력 | 조회 | 출력 | 팝업 | 임베디드 |
| 구분 문자 | C | S | R | P | E |

* 1. 파일(File)명



### 파일명 작성규칙

* 파일명은 반드시 대문자로 시작한다.
* 약어와 약어가 결합하는 경우 뒤 약어의 첫자는 대문자로 시작하도록 한다.
* “-“, ”\_”, ”&”, “ “ 등 특수기호 및 빈 공백은 사용하지 않도록 한다.

### eXbuilder6 파일명

■ 메인화면 파일명 규칙

| **구분** | **내용** | **비고** |
| --- | --- | --- |
| **확장자** | {clx}{html}{jsp} |  |
| **작성규약** | 단위시스템(소문자)+화면유형+약어(대소문자 혼용) |  |
| **규칙** | “XXX[3]”+ “{C}{S}{R}{P}[1]” + “X…X[30]”  C:입력, S:조회, R:보고서, P:팝업, E:임베디드 |  |
| **예** | cmnCCodeMng.clx : 공통 + 입력용(C) + 코드관리화면  apsSPsnSch.clx : 인사 + 조회용(S) + 조회화면  aacEBillDtl.clx : 회계 + 임베디드(E) + 임베디드 화면 |  |

■ UDC 파일명 규칙

학생검색 팝업용 모듈 등 화면의 사용자 정의 컴포넌트로 포함되는 파일로써, eXbuilder6툴에서는 UDC(User Define Control)로 개발된다.

UDC 파일에 대한 명명규칙은 아래와 같다.

- 시작문자는 udc(소문자)로 시작한다.

- [약어]에는 해당 UDC의 목적에 대한 영문약어를 사용하여 명명한다.

- [약어]가 여러 단어로 조합되는 경우, 단어의 시작문자는 대문자로 시작한다.

- 만약, 해당 UDC가 단건 검색 + 팝업 호출용이라면, 파일명은 호출하는 팝업 화면의 파일명에 udc를 붙인 명칭으로 명명한다.

|  |
| --- |
| 명명규칙: udc + 단위시스템[3]+약어(대소문자 혼용)  예시: udcCmnPUser.clx, udcPApsStaff.clx |

### CSS 파일명

* 모든 컨트롤에 대한 디폴트 스타일은 cleopatra Themes에 정의되며, 그 외에 추가적인 상황 및 특징에 맞는 스타일은 Less파일인 others.part.less에 정의하여 사용한다.

### Image 파일명

* 단위시스템 내에서 별도로 필요한 경우 단위시스템사용 디렉토리 내에 위치하도록 한다.

| **구분** | **내용** | **비고** |
| --- | --- | --- |
| **확장자** | gif, png, jpg, bmp |  |
| **작성규약** | image모양을 알수 있게 명명 |  |
| **규칙** | “X…X[30]” |  |
| **예** | arrow.gif : 화살표시 |  |

* 1. eXbuilder6(clx) 소스

### UI 컨트롤

* eXbuilder6 속성뷰의 “ID”항목에 기술한다. 모든 명명규칙은 eXBuilder6 디자이너에서 규정한 네이밍 prefix를 그대로 사용하였으며, 데이터 베이스와 관련된 컨트롤은 “컨트롤 명명규칙” + “데이터베이스의 컬럼ID에 대응하는 이름(또는 약어명)”으로 기술한다. 같은 컨트롤이 여러 개 반복되는 경우에도 반드시 의미있는 이름을 정의하도록 한다.
* 데이터베이스 컬럼ID가 “\_”등의 특수기호가 있는 경우 삭제하도록 하고, 접합단어의 첫자는 대문자로 시작하도록 한다.
* 예시)

\* 컨트롤 타입 : inputbox, 데이터베이스컬럼id : ROLE\_NAME 인 경우 🡺 ipbRoleName

\* 컨트롤 타입 : tree, 데이터베이스컬럼id : DEPT 인 경우 🡺 treDept

| **컨트롤** | **규칙** | **예시** |
| --- | --- | --- |
| **Group** | grp[폼명/그룹명] | grpMain, divSearch |
| **Button** | btn[액션명] | btnSearch, btnSave, btnDelete |
| **Combo** | cmb[컬럼명] | cmbUnitSystemCd |
| **CheckBox** | cbx[컬럼명] | cbxUseYn |
| **ListBox** | lbx[리스트명] | lbxEmpList |
| **InputBox** | ipb[컬럼명] | ipbDeptNm |
| **MaskEditor** | mse[컬럼명] | mseAcctRegNo |
| **NumberEditor** | nbe[컬럼명] | nbeYear |
| **DateInput** | dti[컬럼명] | dtiStDt, dtiEndDt |
| **TextArea** | txa[컬럼명] | txaRemark |
| **TabFolder** | tab[탭명] | tabDept |
| **Image** | img[이미지명] | imgEmpPhoto |
| **RadioButton** | rdb[컬럼명] | rdbGenderCd |
| **Output** | opt[컬럼명] | optTitle |
| **Grid** | grd[그리드명] | grdMessage |
| **Tree** | tre[트리명] | treDept |
| **HTML Object** | htm[컴포넌트명] | htmWebViewer |
| **DataSet** | ds[데이터명] | dsMessage, dsCode |
| **DataMap** | dm[데이터명] | dmParam |
| **Submission** | sub[액션명] | subOnload, subList, subSave |

### 화면 버튼

버튼명은 아래표과 같이 사용한다. 이외의 버튼은 기능을 예측할 수 있는 이름을 사용하도록 한다.

| **버튼명칭** | **버튼 ID명** |
| --- | --- |
| 조회 | btnSearch |
| 신규 | btnNew, btnNewSub |
| 삭제 | btnDelete, btnDeleteSub |
| 저장 | btnSave, btnSaveSub |
| 일괄처리 | btnBatch |

### 서브미션(Submission)

화면에서 서버단에 대한 요청을 위해, 서버단 Controller을 호출 할 때 사용한다.

* “sub” + action name[필수] + 명사[옵션]으로 이루어지며, 접합단어의 첫자는 대문자로 시작한다.
* 동일한 이벤트에 존재하는 Submission이 많은 경우 업무에 종속적인 명사를 사용하지 않고 방향성의미와 일반적인 명사를 사용하도록 한다.
* 이벤트종류가 아닌 특정 컨트롤의 데이터 변경 시 특정컨트롤을 기술하도록 한다.

| **이벤트** | **명명규칙** | **예시** | **비고** |
| --- | --- | --- | --- |
| onBodyLoad | subOnload | subOnload |  |
| list | subList | subList : 조회시 여러 Data군을 조회하는 경우 |  |
| save | subSave | subSaveDtl : 디테일 목록저장Submission |  |
| xxxx | sub + “X…X” | subXxxx | 이벤트는 반드시 동사로 기술한다. |

■ 공통 서비스 이용 서브미션인 경우

조회(S), 출력(R)인 화면에서 onLoad, list 서브미션 호출시에는 각 화면단에서 서브미션 객체를 사용하지 않고, 공통 서비스를 호출하기 때문에 해당 서브미션 명칭은 아래와 같이 규정한다.

명칭은 앞에 기술한 일반적인 경우의 서브미션명칭 + “\_tran” 접미사를 붙인다.

| **이벤트** | **명명규칙** | **예시** | **비고** |
| --- | --- | --- | --- |
| onBodyLoad | subOnload\_tran | subOnload |  |
| list | subList\_tran | subList : 조회시 여러 Data군을 조회하는 경우 |  |
| xxxx | sub + “X…X”\_tran | subXxxx | 이벤트는 반드시 동사로 기술한다. |

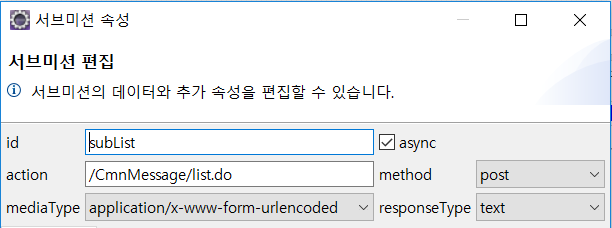
■ 서브미션 URL

- 트랜젝션 Url호출 방식은 정해진 패턴을 사용한다.

- 해당 명명 규칙은 서버 컨트롤러와 밀접한 관계가 있음으로 컨트롤러 명명규칙과 동일하다.

|  |
| --- |
| 액션 URL: **/컨트롤러 클래스에 지정한 RequestMapping URL + /메소드에 지정한 RequestMapping URL** |

|  |
| --- |
| @Controller  @RequestMapping("**/CmnMessage**")  **public** **class** CmnMessageController {    //조회 메소드  @RequestMapping("**/list.do**")  **public** View list(HttpServletRequest request, HttpServletResponse response, RequestData requestData) **throws** Exception {    }  } |



예) /CmnMessage/list.do, /CmnMessage/save.do

|  |
| --- |
| **this.Submit.send(**"subList", null, function(pbSuccess){  if(pbSuccess){  //후처리 로직  }  }**);** |

### 데이터셋

UI 화면내에 디스플레이되는 데이터를 담고 있는 객체에 대한 명명 규칙이다.

데이터셋(DataSet) 객체는 목록성 데이터를 나타내며, 조회조건 파라메터 등과 같은 오브젝트 데이터는 데이터맵(DataMap) 객체를 사용한다.

■ 데이터셋(DataSet)

* ‘ds’로 시작하며, 해당 데이터의 특성을 표현하는 단어를 조합하여 식별이 쉽고 의미있는 명칭을 사용한다. 명명 규칙은 아래와 같다. 접합단어의 첫자는 대문자로 시작한다.

|  |
| --- |
| 명명 규칙: **‘ds’ + 데이터명** |

예) dsMessage, dsDept, dsUnitSystemRcd

* 공통코드를 담는 용도의 데이터셋 명칭은 매칭되는 공통코드에 대한 데이터베이스 테이블 컬럼의 명칭을 기반으로 작성한다.

예) 단위시스템코드(UNIT\_SYS\_RCD)에 대한 데이터셋명은 ‘dsUnitSystemRcd’로 명명함

* 데이터셋 내의 데이터 컬럼은 **영문 대문자**를 기본으로 한다. 즉, 해당되는 데이터베이스 테이블의 컬럼명을 그대로 사용한다.

예)

|  |  |
| --- | --- |
| 데이터셋명 | dsMessage |
| 컬럼명 | UNIT\_SYSTEM\_RCD  MSG\_CD  MSG\_DIV\_RCD  MSG\_TEXT |

■ 데이터맵(DataMap)

* ‘dm’로 시작하며, 해당 데이터의 특성을 표현하는 단어를 조합하여 사용한다. 명명 규칙은 아래와 같다. 접합단어의 첫자는 대문자로 시작한다.

|  |
| --- |
| 명명 규칙: **‘dm’ + 데이터명** |

예) dmParam, dmColumn, dmBatch

* 조회조건을 서버에 보내기 위한 요청 파라메터인 경우에는 데이터맵의 명칭을 dmParam 으로 통일한다.
* 데이터셋 내의 데이터 컬럼은 영문 대/소문자(카멜표기법)를 기본으로 한다.

예)

|  |  |
| --- | --- |
| 데이터맵명 | dsParam |
| 컬럼명 | strUnitSystemRcd  strMessageDivCd  strMessage |

* 1. JavaScript 소스

Java Script를 기술할 때 본 문서에서 기술되지 않은 규칙에 대해서는 일반 Java 소스의 Naming Rule을 적용한다



### 함수(사용자 정의)

- 현재 화면에서 사용할 Java Script 함수(function)를 정의할 때는 함수명 앞에 ‘do’를 붙인다.

예) doSearch() { ... }

- 아래 표의 함수들은 화면마다 반복적으로 쓰일 수 있는 것이므로 같은 용도의 함수를 작성할 경우 아래 표의 함수명을 따르도록 한다. 아래 표에 없는 함수에 대해서는 업무적인 특성을 고려하여 적절한 함수명을 정의하며 함수명은 “do”를 제외하고 자바 메소드명 생성 규칙과 동일하다.

- 함수명 내에 XXX 부분은 관련된 데이터에 대한 업무적인 이름 혹은 대표되는 이름을 기술한다. 대부분 그리드(Grid)에서 맨앞 컨트롤 ID명명규칙을 제외한 id명과 일치한다.

예) doAddXXX : Master에 대한 부서목록(dept)에 대한 그리드 컨트롤은 grdDept이고 함수 이름은 doAddGrdDept가 된다.

- 함수명 내에 XXX가 있는 함수의 경우, 화면에 해당 함수가 하나만 있을 때는 XXX부분을 기술하지 않으며 만일 여러 개일 경우는 그 중에서 가장 대표적인(단축키로 실행할 수 있는) 함수의 XXX부분은 기술하지 않는다.)

| **함수명** | **설명** |
| --- | --- |
| doAddX…X | 일반메인버튼의 ‘추가’ 버튼을 눌렀을 경우 호출되는 함수. |
| doClose | 현재 윈도우 창을 때 호출되는 함수. |
| doDeleteX…X | 일반메인버튼 중에서 ‘삭제’ 버튼을 눌렀을 경우 호출되는 함수. |
| doDetailX…X | 일반메인버튼 중에서 ‘상세’ 버튼을 눌렀을 경우 호출되는 함수. |
| doExcelX…X | 일반메인버튼 중에서 ‘엑셀’ 버튼을 눌렀을 경우 호출되는 함수. |
| doOpenFileX…X | 데이터를 읽어올 파일을 Dialog Box를 통해 찾아서 연다. |
| doSaveFileX…X | 데이터를 파일로 저장한다. |
| doMoveDownX…X | Grid의 ‘위에서 아래로’ 버튼을 누를 경우 호출되는 함수. |
| doMoveLeftX…X | Grid의 ‘우에서 좌로’ 버튼을 누를 경우 호출되는 함수. |
| doMoveLeftAllX…X | Grid의 ‘우에서 좌로 전체이동’ 버튼을 누를 경우 호출되는 함수. |
| doMoveRightX…X | Grid의 ‘좌에서 우로’ 버튼을 누를 경우 호출되는 함수. |
| doMoveRightAllX…X | Grid의 ‘좌에서 우로 전체이동’ 버튼을 누를 경우 호출되는 함수. |
| doMoveUpX…X | Grid의 ‘아래에서 위로’ 버튼을 누를 경우 호출되는 함수. |
| doOnload | 화면 초기화 이후에 추가적인 로직 처리를 위해 호출되는 함수 |
| doUnload | 화면이 닫히기 처리해야 할 내용을 기술한다. |
| doPreviewX…X | ‘미리보기’ 버튼을 눌렀을 경우 사용되는 함수이다. |
| doPrintX…X | ‘인쇄’ 버튼을 눌렀을 경우 호출되는 함수. |
| doRegistX…X | ‘등록’ 버튼을 누를 경우 호출되는 함수. |
| doRetrieveX…X | 검색이나 조회 형태로 읽어오지 않고 일반적으로 데이터 읽어오는 함수. |
| **doSave**, doSaveX…X | 일반메인버튼 중에서 ‘저장’ 버튼을 눌렀을 때 호출되는 함수. |
| **doList**, doListX…X | 검색/조회 버튼을 눌렀을 경우 호출되는 함수. |
| doRestoreX…X | 일반메인버튼 중에서 ‘취소’ 버튼을 누를 경우 호출되는 함수. |
| doUpdateX…X | ‘수정’ 버튼을 누를 경우 호출되는 함수. |
| doValidateX…X | 데이터의 유효성을 검사할 경우 호출되는 함수. |

### 변수

- member : 클래스 내 member변수를 정의 시 사용한다.

- argument : 메소드 및 함수의 argument변수 정의 시 사용한다.

- local : 메소드 및 함수 내 local변수 정의 시 사용한다.

| **구분** | | **Object** | **Control** | **String** | **Number** | **Boolean** | **Date** | **Array** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **oXxxx** | **cXxxx** | **sXxxx** | **nXxxx** | **bXxxx** | **dXxxx** | **aXxxx** |
| member | m | moXxxx | mcXxxx | msXxxx | mnXxxx | mbXxxx | mdXxxx | maXxxx |
| argument | p | poXxxx | pcXxxx | psXxxx | pnXxxx | pbXxxx | pdXxxx | paXxxx |
| local | v | voXxxx | vcXxxx | vsXxxx | vnXxxx | vbXxxx | vdXxxx | vaXxxx |

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