**개발표준정의서(넥사크로)**

|  |  |
| --- | --- |
| 프로젝트명 | 인사급여시스템 |
| 단계 | 설계 |
| 활동 | 개발표준정의 |
| 작업 | 개발표준정의서(UI) 작성 |
| 산출물 | 개발표준정의서(UI) |
| 문서명(파일명) | 넥사크로 개발표준정의서(UI)\_V0.1 |
| 버전 | V0.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | R&R | 성명 | 서명 | 일자 |
| 작성자 | UI꽁통 | 정원식 |  | 2025/05/20 |
| 수정자 |  |  |  |  |

개정 이력

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 버전 | 개정 일자 | 작성자 | 검토자 | 승인자 | 개정 내용 |
| 0.1 | 2025.05.19 | 정원식 |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

목 차

[1. 개요 4](#_Toc198627796)

[1.1 목적 4](#_Toc198627797)

[1.2 적용범위 4](#_Toc198627798)

[1.3 문서의 관리 4](#_Toc198627799)

[2. 개발표준 5](#_Toc198627800)

[2.1 명명규칙 5](#_Toc198627801)

[2.1.1 목적 5](#_Toc198627802)

[2.1.3 서비스 그룹 5](#_Toc198627803)

[2.1.4 공통화면 5](#_Toc198627804)

[2.1.5 업무화면 5](#_Toc198627805)

[2.1.6 컴포넌트 6](#_Toc198627806)

[2.1.7 Dataset ID 7](#_Toc198627807)

[2.1.8 Column ID 7](#_Toc198627808)

[2.1.9 변수 7](#_Toc198627809)

[2.1.10 이벤트 7](#_Toc198627810)

[2.1.11 함수 (Function) 7](#_Toc198627811)

[2.1.12 Scrppt File 8](#_Toc198627812)

[2.2 코딩규칙 10](#_Toc198627813)

[2.2.1 목적 10](#_Toc198627814)

[2.2.2 공통사항 10](#_Toc198627815)

[2.2.3 들여쓰기 10](#_Toc198627816)

[2.2.4 공백 10](#_Toc198627817)

[2.2.5 중괄호 10](#_Toc198627818)

[2.2.6 줄 바꿈 10](#_Toc198627819)

[2.2.7 빈 줄 삽입 11](#_Toc198627820)

[2.2.8 블록문 11](#_Toc198627821)

[2.2.9 조건문 11](#_Toc198627822)

[2.2.10 반복문 12](#_Toc198627823)

[2.2.11 주석 12](#_Toc198627824)

[2.2.12 Script 표준 15](#_Toc198627825)

# 1. 개요

UI 화면개발 시에 지켜야 할 규칙을 정의하여 규격화된 프로그램 개발로 효율성과 가독성을 최대화하여 개발 생산성을 극대화 시킬 수 있으며, 사용자에게 일관된 화면 구성 및 기능을 제공한다.

## 1.1 목적

본 문서는 “인천공항 인사급여시스템 구축” 사업의 성공적인 사업 수행을 위해 구축 대상 시스템의 UI개발가이드를 제공함으로 일관성과 이해성을 높이고, 시스템 설계 및 개발이 체계적이고 정확하게 수행되도록 시스템 개발 시 준수해야 할 UI표준을 제시하는데 그 목적이 있다.

본 문서에서는 nexacro UI 화면을 구현하기 위한 표준 지침을 설명한다.

## 1.2 적용범위

본 문서는 “인천공항 인사급여시스템 구축” 사업의 웹 어플리케이션 UI개발 단계 작업을 수행하는데 적용된다.

## 1.3 문서의 관리

본 문서의 작성 및 관리 책임은 KNFC 디지털융합부에 있으며 본 문서와 관련된 변경사항은 반드시 상기 부서의 승인 하에 이루어 져야 한다.

# 2. 개발표준

## 2.1 명명규칙

명명규칙은 컴퓨터가 효율적으로 실행 될 수 있도록 코드를 만들어야 한다는 목표와 인간이 쉽게 이해할 수 있는 코드를 만들어야 한다는 서로 상층되는 두가지 목적을 만족 시키기 위한 것 입니다.

### 2.1.1 목적

(1) 일정한 규칙을 가지고 이름을 부여함으로써 개발자공통의 의사소통이 가능함으로 분산 개발이 가능하고 의사소통이 원활 해 진다.

(2) 일정한 규칙을 가지고 이름을 정함으로써 다른 개발자가 운영하는데 있어 혼란을 최소화 할 수 있고 재사용이 가능 해 진다.

(3) 유지보수를 쉽게 이해하고 추측 가능하므로 시스템의 관리 및 운영의 편리성을 제공한다.

### 2.1.3 서비스 그룹

(1) 영문 카멜을 사용한다.

(2) 서비스 그룹은 업무코드와 동일하게 명명 한다

(3) 하위 서비스 그룹 생성 시 상위 서비스 그룹 코드를 포함 해야 한다

|  |  |  |
| --- | --- | --- |
| 모듈코드 | 그룹코드 | 비고 |
| view | view |  |

### 2.1.4 공통화면

(1) 공통화면의 경우 prefix(업무코드) + 의미 있는 단어로 이름을 부여한다.

(2) id는 카멜표기법(camel case)을 사용한다

|  |  |  |
| --- | --- | --- |
| 화면유형 | 예제 | 비고 |
| 알림 | cmmAlert.xfdl | 알림 창 팝업 |
| 확인 | cmmConfirm.xfdl | 메시지 확인 팝업 |

### 2.1.5 업무화면

(1) 영문 소문자를 사용한다.

(2) 명명 규칙은 [서비스 그룹] + [일련번호 4자리] + [화면분류 2자리] 로 명명 한다.

(3) 일련번호는 0001 ~ 9999까지 사용 가능하다.

|  |  |  |  |
| --- | --- | --- | --- |
| 화면유형 | 화면분류 | 예제 | 비고 |
| Form | fm | cma0001fm.xfdl | 거래처 관리 |
| Popup | pu | cma0001pu.xfdl | 거래처 검색 팝업 |
| Layer, Tabpage | ly | Cma0001ly.xfdl | 거래처 상세 정보 |

### 2.1.6 컴포넌트

(1) 컴포넌트는 카멜표기법(camelCase)로 작성하며, 컴포넌트 유형에 따라 Prefix 사용 한다.

(2) 명명 규칙은 [컴포넌트 Prefix] + [의미 있는 단어]로 명명 한다.

(3) [의미 있는 단어]는 용어사전을 준수하여 용어명에 맞는 영문을 사용한다.

(4) Dataset Column과 바인딩 되는 경우에는 관련된 Dataset Column Name을 바탕으로 지정하는 것을 원칙으로 한다.

*[예시] Dataset Column 명이 CODE\_NAME인 경우 화면 항목명은 edtCodeName*

(5) Script에서 사용하지 않는 중복되는 컴포넌트는 Prefix ID에 00~99를 붙여서 사용한다.

*[예시] Static : sta01, sta02*

인쇄 : btnPrint (컴포넌트 id) / fnPrint (인쇄 함수)

|  |  |  |
| --- | --- | --- |
| 컴포넌트 | 접두어(Prefix) | 예제 |
| Button | btn | btnSave, btnSearch, … |
| Calendar | cal | calFor, … |
| CheckBox | chk | chkAll, … |
| ComboBox | cbo | cboAccItem, … |
| Dataset | ds | dsUser, dsMenu, … |
| Division | div | divUserInfo, … |
| Edit | edt | edtUserNm, … |
| Gird | grd | grdClientList, … |
| ImageViewer | Img | imgTitle, … |
| ListBox | lst | lstMenu, … |
| MaskEdit | mae | maeAmount, … |
| PopupDiv | pdiv | pdivMemo, … |
| Radio | rdo | rdoYn, … |
| Shape | shp | shpBox, … |
| Spin | spn | spnAddVal, … |
| Static | sta | staName, … |
| Tab | tab | tabClient, … |
| TabPage | tpg | tpgSearch, … |
| TextArea | txa | txaMemo, … |
| TreeView | tv | tvMenu, … |
| GroupBox | grb | grbEtc, … |
| MenuBar | mnb | mnbTopMenu, … |
| ProgressBar | pgb | pgbUpload, … |
| PopupMenu | pmn | pmnExtGrid |

### 2.1.7 Dataset ID

(1) Dataset의 ID는 Service에 정의된 Param Dataset, Result Dataset의 명칭으로 사용한다

(2) input Param Dataset 명칭은 “dsInput” 명칭으로 사용을 권장한다.

(3) output Param Dataset 명칭은 “dsOutput” 명칭으로 사용을 권장한다.

### 2.1.8 Column ID

(1) Column 명은 DB의 Column명과 동일하게 사용한다

### 2.1.9 변수

(1) 기본적으로 카멜표기법(camel case)을 사용한다.

(2) 명명 규칙은 [용어사전] 을 조합하여 30자 이내로 구성 한다.

|  |  |  |
| --- | --- | --- |
| Data Type | 약어 | 적용예시 |
| Number | n | nCnt |
| String | s | sValue |
| Boolean | b | bResult |
| Object | obj | objGrid |
| Array | arr | arrMessage |

(3) 로컬변수는 [데이터 타입(선택)] + [의미 있는 단어] 로 명명 한다.

*[예시] var nCnt = 0;*

(4) 화면 전역변수는 [접두어 fv] + [데이터 타입(선택)] + [의미 있는 단어]로 명명 한다.

*[예시] this.fvCustName = “”;*

(5) 글로별 변수는 Environment Variables과 AppVariables의 Variables와 Datasets를

의미한다. 화면 내의 변수와 구분하기 위해 Environment Variables는 “ev”로 시작하고,

AppVariables의 Variables는 “gv”로 시작하며, AppVariables의 Datasets “gds”로 시작한다.

*[예시] Environment Variables = evRunMode;*

*AppVariables Variables = gvUserId*

*AppVariables의 Datasets = gdsMessage*

(6) loop index에서 사용하는 변수는 i, j, k, l, x, y, z등을 (관용적으로) 사용할 수 있다.

### 2.1.10 이벤트

(1) 임의의 이벤트 명을 부여하지 않는다.

(2) 명명 규칙은 컴포넌트의 이벤트 속성 창에서 해당 이벤트 더블클릭으로 자동 명명 한다.

*[예시] this.btnSearch\_onclick = function(obj:nexacro.Button, e:nexacro.ClickEventInfo) { }*

### 2.1.11 함수 (Function)

(1) 명명 규칙은 [접두어] + [동사] 로 명명 한다.

(2) [동사의] 첫 글자는 대문자를 사용하며, 숫자 및 특수문자를 사용하지 않는다.

(3) 표준함수의 경우 반드시 아래 명칭의 함수를 사용 한다.

예)

신규 : this. fnNew\_onclick = function(obj,e) { }

등록 : this. fnSearch\_onclick = function(obj,e) { }

수정 : this. fnUpdate\_onclick = function(obj,e) { }

삭제 : this. fnDelete\_onclick = function(obj,e) { }

저장: this. fnSave\_onclick = function(obj,e) { }

인쇄 : this. fnPrint\_onclick = function(obj,e) { }

| 로컬함수 | | 공통함수 | |
| --- | --- | --- | --- |
| 접두어 | 예제 | 접두어 | 예제 |
| Fn | fnSet | gfn | gfnIsNull |

### 2.1.12 Scrppt File

(1) Script 파일명은 프로젝트에서 공통으로 사용하는 파일과 각 업무에서 사용하는 파일로 구분하여 정의한다.

(2) 프로젝트에서 공통으로 사용하는 공통함수는 기능별로 관리되며 기능에 해당하는 의미 있는 단어로 이름을 이름을 부여하고, 프로젝트 소스 내 \_resource\_/LibCommon/ 폴더에 위치하고 nexacro 엔진 load 시 loading 되며, form의 prototype에 추가되어 모든 form에서 실행되게 된다.

| No | 파일명 | 파일의 기능 |
| --- | --- | --- |
| 1 | frame.js | 프레임 관련 공통 함수 |
| 2 | util.js | Date, String 관련 공통 함수 |
| 3 | transaction.js | Service 관련 공통 함수 |
| 4 | popup.js | 팝업 관련 공통 함수 |
| 5 | language.js | 다국어 관련 공통 함수 |
| 6 | file.js | File 업/다운로드 관련 공통 함수 |
| 7 | message.js | 메세지 처리 관련 공통 함수 |
| 8 | grid.js | 그리드 관련 공통 함수 |
| 9 | Excel.js | 엑셀 import/export 관련 공통 함수 |

3) 화면에서 script를 include하는 것은 성능저하를 발생할 수 있어 사용을 제한하며 부득이한 경우에는 [prefix 모듈코드] + [의미 있는 단어]로 명명 한다.

*[*

## 2.2 코딩규칙

규격화된 프로그램 소스는 효율성 및 가독성을 증대시키고 유지 보수의 비용을 절감할 수 있다. 주석은 어디에 어떤 방법으로 작성하고, 괄호의 위치는 어떻게 하며, 들여쓰기 공간의 크기와 표현 등으로 나누어 정의 한다.

### 2.2.1 목적

(1) 코딩 규칙은 소스개발에 대한 규칙을 정의하는 것으로 표준화와 일관성을 확보하는 것을 목적으로 한다.

(2) 코드의 일관성이 유지되고 예측가능 해지며 읽고 이해하기가 쉬워지므로 다른 개발자들과 함께 유지보수가 가능해 진다.

### 2.2.2 공통사항

(1) 코드는 항상 읽기 쉽고 개발자들이 이해할 수 있게끔 작성해야 한다..

(2) 최적화가 필요하다면 독립적인 모듈로 작성해야 한다.

(3) 주석은 모든 코드에 상세히 기술하는 것을 원칙으로 한다.

(4) 소스 코드는 불가피한 내용을 제외하고 원칙적으로 중복을 금지한다.

(5) 개발 및 테스트를 수행함에 있어 기능과 성능은 물론 보안에도 각별한 주의를 기울인다..

### 2.2.3 들여쓰기

(1) 모든 내용을 들여쓰기를 적용하며 indent는 4칸 (nexacro 기본 Tab Size)으로 적용한다.

### 2.2.4 공백

(1) 한 줄에는 하나의 statement만 기술한다.

(2) 구성요소 사이에 공백은 1칸 공백을 기본으로 한다.

(3) 단항(unary) 연산자는 공백을 두지 않는다.

(3) 괄호 안에 괄호가 있는 경우에는 괄호 사이에 공백을 두지 않는다.

### 2.2.5 중괄호

(1) 쌍을 이루는 괄호는 반드시 줄을 맞춘다.

(2) 중괄호에 주석을 기입하는 경우 ‘//’ 주석을 사용한다

(3) ‘{’는 기존의 ‘{‘와 비교해서 공백(4칸)을 준다.

(4) ‘}’는 짝이 되는 ‘{‘과 동일하게 공백(4칸)을 준다.

### 2.2.6 줄 바꿈

(1) 한 줄인 경우에도 {}로 묶는다.

그러나 if (a < b) return a; 과 같이 한 줄에 쓰는 경우에는 { }로 묶지 않아도 된다.

*[예시] Argument가 많아 한 줄에 쓸 수 없거나 가독성이 떨어질 경우*

*this.fnOpen = function (sPrefix, sFormID, sInArgument, iWidth, iHeight,*

*sOpenStyle, iLeft, iTop)*

*{*

*This.fnCall();*

*}*

*[예시] 조건, Dataset Argument 등으로 사용할 String을 연결할 경우 (가독성이 떨어지는 경우)*

*this.fnClick = function (objBtn)*

*{*

*var outDa = “”;*

*var params = " initMsgList=T"*

*+ " userId=69709"*

*+ " custNm=KT"*

*+ " rnsCnfmYn=Y"*

*}*

*[예시] 조건이 복잡하고 많아 가독성이 떨어지는 경우 (조건별개 행 및 줄 맞춤)*

*if ( (condition1 && condition2) ||*

*(condition3 && condition4) ||*

*(condition5 && condition6)*

*)*

*{*

*statements;*

*}*

### 2.2.7 빈 줄 삽입

(1) 빈 줄에 대한 규정은 없으나 가독성을 위하여 블록 이후, 성격이 다른 경우, 기타 가독성

을 위한 빈 경우 빈 줄 삽입을 권장한다.

### 2.2.8 블록문

(1) 내용이 없는 블록문은 사용하지 않는다.

(2) 블록문은 다른 문들과는 다르게 끝에 세미 콜론(;)을 붙이지 않는다.

### 2.2.9 조건문

(1) 조건문은 3중을 초과해서 사용하지 않는 것을 원칙으로 한다.

(2) if 문은 다음과 같이 사용한다.

*if (condition)*

*{*

*statements;*

*}*

*else if (condition)*

*{*

*statements;*

*}*

*else*

{

statements;

}

(3) if 문에 else (또는 else if)가 포함될 경우 가독성을 위해 아래와 같은 형태도 가능하다.

*if (condition) {*

*statements*

*}*

*else {*

*statements*

*}*

### 2.2.10 반복문

(1) 반복문 내는 가급적이면 신규 변수를 생성하지 않는다.

(2) 반복문 조건절에는 연산이 들어가지 않도록 한다.

(3) for문과 while, switch문은 다음과 같이 사용한다.

*//for문*

*for (var i = 0; i < iCnt; i++)*

*{*

*statements;*

*}*

*//while문*

*while (i > iCnt)*

*{*

*statements;*

*}*

*//switch*

*switch (svcID)*

*{*

*case "search":*

*statements;*

*break;*

*default:*

*break;*

*}*

### 2.2.11 주석

(1) 파일 헤더 주석.(/title header)

1) 모든 파일의 맨 상단에 기술하며 아래와 같이 작성한다

2) 파일의 주요 내용이 변경 될 때는 주석의 History 영역에 date, Modifier, Description을 기록한다.

*/\*\**

*\* @MenuPath Sample 화면*

*\* @FileName work01.xfdl*

*\* @Creator ex) 홍길동*

*\* @CreateDate 2023/02/28*

*\* @Desction 샘플 부서코드 관리*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* date Modifier Description*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* 2023/02/28 ex) 홍길동 최초 생성*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\*/*

(2) 공용함수 주석.

1) 주석의 종류는 class, param, return를 반드시 작성한다.

2) jsdoc 에서 지정한 주석 표준을 사용한다.

*/\*\**

*\* @class : 레이블 변경 함수 – 다국어*

*\* @param : sLocale {String} - Local언어*

*\* @param : objForm {Form} – Form*

*\* @return : N/A*

*\*/*

| 공통함수 주석 세부 항목 | 설명 |
| --- | --- |
| @class | 함수설명(jsdoc으로 표현하기 위해 공통함수는 @class를 사용함, 일반함수는 @description 사용) |
| @param | Parameter: {Data Type} param명 - param설명 |
| @return | Return값: {Data Type} return설명 |

(3) 일반함수 주석(/function)

1) 함수는 param, return까지 작성을 권장한다.

2) 이벤트 함수는 param과 return을 생략하며 @description에 기능을 설명한다.

*/\*\**

*\* @description : Transaction CallBack 함수*

*\* @param : {String} sId - transaction을 구분하기 위한 id값*

*\* @param : {Number} nErrorcode - Error Code*

*\* @param : {String} sErrorMsg - Error 메시지*

*\* @return : N/A*

*\*/*

| 함수 및 이벤트 주석 세부 항목 | 설명 |
| --- | --- |
| @description | 함수설명 |
| @param | Parameter: {Data Type} param명 - param설명 |
| @return | Return값: {Data Type} return설명 |

(4) 화면 내 일반함수 주석표준

| 구분 | 설명 |
| --- | --- |
| 함수 및 주석 처리 표준 | 공통함수 주석  /\*\*  \* @class : Transaction 공통 CallBack 함수  \* @param : {String} sId - transaction을 구분하기 위한 id값  \* @param : {Number} nErrorcode - Error Code  \* @param : {String} sErrorMsg - Error 메세지  \* @return : N/A  \*/  화면 내 일반함수 주석  /\*\*  \* @description : Transaction CallBack 함수  \* @param : {String} sId - transaction을 구분하기 위한 id값  \* @param : {Number} nErrorcode - Error Code  \* @param : {String} sErrorMsg - Error 메세지  \* @return : N/A  \*/  이벤트 주석  /\*\*  \* @description화면 로드 시 이벤트  \*/ |
| 여러 라인주석: 함수 중간에 여러 줄의 주석을 사용할 필요가 있을 경우 | /\*  \*/ |
| 주석의 indentation 방식 | /\* indentation 은 설명하고자 하는 부분과 일치시킴 \*/  if (flag == 1)  {  statements  }  /\* 위 줄이 비워 있으므로 한 줄 띄지 않고 주석 작성 가능 \*/  else  {  statements  } |

### 2.2.12 Script 표준

- 업무폼 : /base script

- 팝업 : /base script poup

(1) 모든 화면의 Script의 주석, 기능별 함수의 위치, 공통기능 함수명 등의 표준화를 통해 개발 및 운영 시 효율성과 가독성을 최대화 하여 개발 생산성을 극대화 한다..

1) 모든 파일의 맨 상단에 기술하며 아래와 같이 작성한다

2) 파일의 주요 내용이 변경 될 때는 주석의 History 영역에 date, Modifier, Description을 기록한다.

*/\*\**

*\* 타이틀*

*\*@FileName test005*

*\*@Creator 홍길동*

*\*@CreateDate 2025/05/15*

*\*@Desction*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 소스 수정 이력 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* date Modifier Description*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\**

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\*/*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* Form Event*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*/\*\**

*\* @description 화면 onload시 처리내역(필수)*

*\*/*

*this.addEventHandler("onload", function(obj,e)*

*{*

*},this);*

*/\*\**

*\* //paging 버튼 // next prev 클릭 이벤트 후처리*

*\* @param {number} 누른 페이지*

*\* @param {number} 시작데이터rownum계산*

*\* @return*

*\* @example*

*\* @memberOf*

*\*/*

*this.cfnPaging\_onclick = function(nPage,nRecordOffset)*

*{*

*trace(" ## nPage : " +nPage + "<> nRecordOffset : " + nRecordOffset);*

*}*

*//popup titlebar close(x) 버튼 클릭 event*

*this.cfnTitlebarClose\_onclick = function()*

*{*

*return false;*

*}*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* CALLBACK 콜백 처리부분(Transaction, Popup)*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*this.fnCallback = function(id,code,msg)*

*{*

*// 에러 시 화면 처리 내역*

*if(code < 0) return;*

*switch(id) {*

*case "":*

*break;*

*default:*

*}*

*};*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* CRUD 및 TRANSACTION 서비스 호출 처리*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* 사용자 FUNCTION 영역*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*\* 각 COMPONENT 별 EVENT 영역*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*