

DevPACK

개발 표준 및 개발 가이드

TOBESOFT

목차

DevPACK	1
개발 표준 및 개발 가이드.....	1
파트1. 개요.....	4
파트2. 시스템 환경.....	4
1. 넥사크로.....	4
1.1 프로그래밍 언어.....	4
1.2 개발 환경	5
1.3 실행 환경	5
1.4 배포 환경	6
1.5 시스템 요구사항.....	6
2. 서버.....	8
2.1 넥사크로 X-API.....	8
2.2 넥사크로 Xeni	9
2.3 넥사크로 서버 라이선스	11
파트3. 개발 표준	12
1. 명명 표준.....	12
1.1 화면 FORM FILE (.xfd)	12
1.2 Script File (.xjs or .js).....	12
1.3 Component ID	13
1.4 Function (함수명).....	16
1.5 Variable (변수).....	17
2. 코딩 표준	19
2.1 기본 사항	19
2.2 줄 바꿈	19

2.3 빈 줄 삽입	20
2.4 Block Statement	20
2.5 Comments (주석).....	21
파트4. 개발 가이드.....	24
3. 프로젝트 구성	24
3.1 UI 프로젝트 폴더 구조	24
3.2 Nexacro Studio 설정	26
3.3 Environment Variables 와 Application Variables	34
3.4 Global Dataset	36
3.5 공통 Library	37
3.6 Frame 구조	38
4. 기능별 개발 방법	42
4.1 Transaction	42
4.2 Grid 기능.....	44
4.3 Excel Export / import	46
4.4 팝업 처리	48
4.5 메시지	51
4.5 정합성 (Validation) 체크	53
4.6 공통버튼 사용	55
4.7 공통버튼 추가버튼	56

파트1. 개요

넥사크로플랫폼 17.1은 하나의 소스로 웹과 모바일, 데스크톱 앱을 디자인, 개발, 배포할 수 있는 소프트웨어 개발 플랫폼입니다. 기업 내에서 업무에 따라 웹브라우저에서 바로 접근할 수 있도록 운영하다가 윈도우 운영체제 환경에서 특정 장비와 연동해야 할 때에는 해당 어댑터 기능만 추가해서 윈도우 앱 형태로 구현하고 배포할 수 있습니다. 개발부터 배포까지 하나의 도구를 사용해 다양한 사용자의 실행 환경과 요구에 대응할 수 있습니다.

DevPack은 넥사크로플랫폼 17.1버전으로 만들어진 넥사크로플랫폼 프로젝트로 UI프레임, 다양한 샘플 및 기본적인 시스템 관리 기능을 제공합니다.

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

파트2. 시스템 환경

1. 넥사크로

1.1 프로그래밍 언어

넥사크로플랫폼은 다른 프로그래밍 언어와 달리 사용자에게 보이는 화면을 정의하는 부분과 비즈니스 로직을 처리하는 스크립트로 구분된다. 또한 화면에 원하는 디자인을 적용 하기 위해 스타일과 테마를 적용할 수 있는 기능을 제공한다.

화면을 배치하는 부분은 XML 기반으로 각 컴포넌트의 속성과 바인딩, 이벤트 등의 정보를 관리한다. 다양한 실행 환경을 지원할 수 있도록 MLM(Multi Layout Manager) 기능을 지원하며 관련된 속성을 사용할 수 있다. 앱 실행에 필요한 환경 정보는 별도의 파일에서 관리한다.

구분	파일명(확장자)	용도
프로젝트	*.xprj	<ul style="list-style-type: none">• 프로젝트 정보• Environment 파일 경로• TypeDefinition 파일 경로• AppVariables 파일 경로• Application Information 파일 경로

구분	파일명(확장자)	용도
Environment	environment.xml	<ul style="list-style-type: none"> • 실행 환경 정보 • themeid • Screen 정보 (Screen 정보 설정 시 실행 환경 정보에서 설정한 값을 덮어쓸 수 있습니다). • variables • cookies
TypeDefinition	typedefinition.xml	<ul style="list-style-type: none"> • 모듈 • 컴포넌트 • 서비스 • 프로토콜 • 업데이트
initValue	*.xiv	<ul style="list-style-type: none"> • 컴포넌트 또는 오브젝트 속성 초기값 설정
Application Information	*.xadi	<ul style="list-style-type: none"> • Application 정보 • screenid • 프레임 속성
AppVariables	appvariables.xml	<ul style="list-style-type: none"> • Application 내에서 공유하는 공통 변수
Form	*.xfdl	<ul style="list-style-type: none"> • 화면 레이아웃 • 화면 폼 속성 • 컴포넌트 속성 • 추가 레이아웃 • 스크립트

넥사크로플랫폼에서 작성한 코드는 빌드 과정을 거쳐 자바스크립트 기반의 코드로 변환된다. 실제 실행 환경에서는 변환된 자바스크립트 코드를 실행하게 된다.

1.2 개발 환경

넥사크로플랫폼은 위지윅(WYSIWYG) 기반의 개발 툴인 넥사크로 스튜디오를 제공한다. 넥사크로 스튜디오 내에서 실행 환경과 상관없이 앱을 개발할 수 있으며 생성된 코드는 넥사크로플랫폼 프로그래밍 언어로 저장된다.

넥사크로 스튜디오는 마이크로소프트 윈도우 운영체제만을 지원하지만 개발된 앱은 어떤 운영체제나 어떤 디바이스든 상관없이 최적화된 사용 환경으로 배포할 수 있다.

앱 빌더를 사용하면 IOS/iPadOs, macOS, 안드로이드 운영체제를 지원하는 앱을 간단한 설정만으로 생성하고 배포할 수 있다.

1.3 실행 환경

넥사크로플랫폼은 넥사크로 프레임워크를 기본으로 앱이 실행되며 각 실행 환경에 따

라 최적화된 구조를 제공한다. 앱 실행 환경에 따라 데스크탑, 모바일 환경으로 나누고 앱 실행 방식에 따라 NRE(Nexacro Runtime Environment), WRE(Web Runtime Environment)으로 구분한다. 앱 실행 방식에 따라 추가된 기능을 사용할 수 있다. 예를 들어 안드로이드와 iOS/IpadOS 운영체제의 NRE 버전은 카메라, 주소록, SMS 등 모바일 디바이스와 연동된 추가 기능을 사용할 수 있다.

1.4 배포 환경

배포란 앱이 실행하는데 필요한 자원을 클라이언트에 설치하는 일련의 작업을 의미한다. 사용 환경에 따라 넥사크로플랫폼에서 개발된 앱과 필요한 모듈을 내려 받아 클라이언트에 설치하게 된다.

넥사크로플랫폼 앱은 사용자가 사용하는 클라이언트에서 동작한다. 하지만 데이터 처리와 같은 작업을 위해 WAS(Web Application Server)를 필요로 할 수 있다. 넥사크로플랫폼은 데이터 처리를 위한 X-API모듈과 함께 제공하고 있다. 또한 필요에 따라 데이터를 실시간으로 처리해야 한다면 X-Push와 같은 추가적인 기술을 사용할 수 있다.

기본 배포 작업은 HTTP 프로토콜을 사용한다. 하지만 인터넷 접속을 지원하지 않는 환경에서는 앱 실행에 필요한 자원을 별도로 매체로 배포해 사용한다.

1.5 시스템 요구사항

1.5.1 Web Runtime Environment

웹 브라우저	최저 사양	최고 사양
Google Chrome	59.0.3071	최신버전
Firefox	52.0	최신버전
Internet Explorer	8.0	11.0
Edge	42.17134.1.0	최신버전
Opera	11.0	최신버전
Safari	4	최신버전

1.5.2 Nexacro Runtime Environment

항목	사양	windows	Macos
CPU	최저	인텔 펜티엄 4 1.4GHz	
	권장	인텔 코어 i5 7세대 이상	
메모리	최저	1GB	
	권장	8GB	
HDD(ROM)	최저	8GB	
	권장	128GB	
플랫폼		Windows Vista Home Premium SP2 이상 Windows 10 V2004 이하	macOS 10.10.5 이상 macOS 10.14.4 이하
		Windows Server 2008 R2 SP1 이상 Windows Server 2016 이하	

1.5.3 Mobile Web Runtime Environment

모바일 웹 브라우저	최저 사양	최고 사양
Android 기본 브라우저	Android 5.0	Android 9
Android 구글 크롬 브라우저	59.0.3071	최신버전
iOS 기본 브라우저	iOS 9	iOS 13.3
iPadOS 기본 브라우저		iPadOS 13.3

1.5.2 Mobile Nexacro Runtime Environment

항목	사양	Android	IOS/Ipad OS
CPU	최저	ARM Cortex-A8 600 MHz	iOS 9 이상 지원 기기
	권장	ARM Cortex-A8 1GHz이상	
디스플레이	최저	HVGA	
	권장	HVGA이상	
메모리	최저	3GB	
	권장	4GB 이상	
HDD(ROM)	최저	32GB	
	권장	64GB 이상	
플랫폼		Android 5.0 (API 21) 이상 Android 9.0 (API 28) 이하	iOS 9 이상 iOS/iPadOS 13.3 이하

2. 서버

2.1 넥사크로 X-API

넥사크로플랫폼 X-API는 서버 모듈로 제공되며 서버와 클라이언트 간에 데이터를 위해 필요한 기능을 제공한다. 기본적인 데이터 송신, 수신 기능과 간단하게 데이터를 가공하는 기능을 제공해 데이터 처리 과정을 단순화 한다.

2.1.1 설치

X-API는 자바 기반 서버 모듈로 제공되면 JDK 또는 JRE 1.4 이상 버전이 필요하다.

제공받은 X-API.jar 파일을 WAS의 /WEB-INF/lib 디렉토리 또는 정의된 클래스 경로에 복사한다.

파일명	필수여부	설명	참조
nexacro-xapi-x.x.x.jar	Y	X-API	
commons-logging-x.x.x.jar	Y	X-API 내부 로깅	Apache Commons Logging

2.2 넥사크로 Xeni

Nexacro-xeni는 서버 모듈로 제공되며 클라이언트에서 파일 형태로 데이터 처리를 할 때 필요한 기능을 제공한다. 파일에 포함된 데이터를 그리드 컴포넌트로 가져오거나 그리드에 연결된 데이터를 파일로 내보내는 기능을 처리한다.

제니는 아래와 같은 파일 형태와 기능을 지원한다.

파일 형태	확장자	기능
Microsoft Excel 97-2003	xls	export / import
Microsoft Excel 2007/2010	xlsx	export / import
CSV format file	csv	import
HancomOffice Hancell 2010	cell	export
HancomOffice Hancell 2014	cell	export / import

2.2.1 설치

nexacro-xeni는 자바 기반 서버모듈로 제공되며 버전에 따라 아래와 같이 지원 환경 및 기능이 달라진다.

Nexacro-xeni	JDK version	POI 라이브러리 버전	유지보수
1.0.0	1.5 이상	POI 3.10	버그 수정만 지원
2.x.x	1.8 이상	POI 4.1.2	신규, 변경 포함

제공된 war(Web application ARchive) 파일을 직접 배치하거나 war 파일의 압축을 풀어 필요한 파일을 WAS의 /WEB-INF/lib 디렉터리 또는 정의된 클래스 경로에 복사해 사용할 수 있다.

War 파일을 직접 배치하지 않고 압축을 풀어 복사하는 경우에는 라이브러리 버전 차이로 문제가 발생할 수 있다.

Nexacro-xeni가 동작하려면 X-API가 설치되어 있어야 한다.

기존 컨텍스트에 파일을 복사한 경우에는 아래 내용을 web.xml에 추가해야 한다.

```
<servlet>
    <servlet-name>XExportImport</servlet-name>
    <servlet-
class>com.nexacro17.xeni.services.GridExportImportServlet</servlet-
class>
</servlet>
<servlet>
    <servlet-name>XImport</servlet-name>
    <servlet-
class>com.nexacro17.xeni.services.GridExportImportServlet</servlet-
class><class>
</servlet>

<servlet-mapping>
    <servlet-name>XExportImport</servlet-name>
    <url-pattern>/XExportImport</url-pattern>
```

```

</servlet-mapping>
<servlet-mapping>
    <servlet-name>XImport</servlet-name>
    <url-pattern>/XImport</url-pattern>
</servlet-mapping>

<context-param>
    <param-name>export-path</param-name>
    <param-value>/export</param-value>
</context-param>
<context-param>
    <param-name>import-path</param-name>
    <param-value>/import</param-value>
</context-param>
<context-param>
    <param-name>monitor-enabled</param-name>
    <param-value>true</param-value>
</context-param>
<context-param>
    <param-name>monitor-cycle-time</param-name>
    <param-value>30</param-value>
</context-param>
<context-param>
    <param-name>file-storage-time</param-name>
    <param-value>10</param-value>
</context-param>

```

2.3 넥사크로 서버 라이선스

제공받은 라이선스 파일(nexacro17_server_license.xml)파일을 X-API및 Xeni jar파일과 같은 디렉토리 또는 정의된 클래스 경로에 복사하여 둔다.

1개 이상의 라이선스 파일이 다른 경로로 복사된 경우 jar파일과 같은 디렉토리에 있는 라이선스 파일을 먼저 적용한다.

서버 환경 설정에 따라 jar파일과 라이선스 파일을 복사하고 WAS 재시작이 필요 할 수 있다.

파트3. 개발 표준

1. 명명 표준

1.1 화면 FORM FILE (.xidl)

1.1.1 공통화면

공통화면의 경우 prefix[업무분류] + 의미 있는 단어의 조합으로 부여한다.

예] cmm (공통화면) + 메시지 : cmmAlert.xidl

1.1.2 업무 화면

업무 화면은 biz 하위에 의미 있는 단어의 조합으로 폴더로 구성한다.

폴더 prefix는 biz +의미 있는 단어의 조합으로 부여한다.

예] biz + 업무 기본 : bizBase

파일명은 biz 하위 폴더 명 + 의미 있는 단어의 조합 + 화면 분류 코드(자리)로 정의한다.

화면	유형	코드	예제	비고(예제)
업무 화면	Form	FM	baseBoardMgFm.xidl	게시판 관리 화면
	Popup	PU	baseBoardInputPu.xidl	게시판 입력 팝업
	Tabpage	TB	baseCommonCodeTb.xidl	공통 코드 관리 탭 연결 화면
	Div	DV	baseCommonCodeDv.xidl	공통 코드 관리 디비전 연결 화면

표 1-1 업무 화면 유형 구분 코드 예시

1.2 Script File (.xjs or .js)

1.1.2 전체 공통 Script File

프로젝트에서 공통으로 사용하는 공통 함수는 기능별로 관리되며, 기능에 해당하는 의미 있는 단어로 이름을 부여하고, 프로젝트 소스 내 nexacro18lib/component/DevPackLib/ 폴더 하위에 위치하고 nexacro 엔진 load

시 함께 로딩된다. 해당 라이브러리는 form의 prototype 형태로 추가되어 모든 form에서 form base로 실행 되게 된다.

NO	파일명	파일의 기능	비고
1	Frame.js	프레임 관련 공통 함수	
2	Util.js	Date, String 관련 공통 함수	
3	Transaction.js	Service 관련 공통 함수	
4	Validation.js	Vaidation 관련 공통 함수	
5	Popup.js	팝업 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	
10	ExtLib.js	기타 업무 베이스 공통 함수	

표 1-2 공통 Script 파일 목록

1.1.3 업무 공통 Script File

화면에서 script를 include하는 것은 성능 저하를 발생할 수 있어 사용을 제한하며, 부득이한 경우biz하위 폴더명 + 의미 있는 단어 + Lib 이름을 부여하며, 해당 업무 폴더에 위치한다.

예] bizBase업무에서 게시판 공통 library js : baseBoardLib.xjs

1.3 Component ID

화면의 컴포넌트 ID는 컴포넌트의 PrefixID + 의미있는 단어로 이름을 부여한다.

- 1) Dataset Column과 바인딩 되는 경우에는 관련된 Dataset Column Name을 바탕으로 지정하는 것을 원칙으로 한다.

예] Dataset Column 명이 CODE_NM인 경우 화면 항목명은 edtCodeNm

2) Script에서 사용하지 않는 중복되는 컴포넌트는 Prefix ID 에 00~99를 붙여서 사용한다.

예] Static : sta00, sta01 ..

Component	Prefix	예제	비고
Button	btn	btnSave	
Combo	cbo	cboDept	
Edit	edt	edtName	
MaskEdit	msk	mskCusNo	
TextArea	txa	txaRemart	
Static	sta	staName	
Div	div	divMain	
PopupDiv	pdiv	pdivSub	
Radio	rdo	rdoPersontype	
Checkbox	chk	chkYn	
Listbox	lsb	lsbCatalog	
Grid	grd	grdMain	
Spin	spn	spnAge	
Menu	mnu	mnuFill	
PopupMenu	pmnu	pmnuSub	
Tab	tab	tabDetail	

Component	Prefix	예제	비고
Groupbox	grb	grbGubun	
Calendar	cal	calFromTo	
ImageViewer	img	imgGraph	
Progressbar	pgb	pgbSale	
Plugin	plg	plgReport	
Dataset	ds	dsList	
WebBrowser	web	webHomepage	
FileDialog	fdg	fdgPop	
File upload	ful	fulList	
FileUpTransfer	fut	futFile	
File download	Fdd	fdlList	
FileDownTransfer	Fdt	fdtFile	
VirtualFile	vtf	vtfFile	
Sketch	skc	skclmg	

표 1-3 Component ID 분류

1.3.1 Dataset 및 Column

Dataset의 ID는 Service에 정의된 Param Dataset, Result Dataset의 명칭으로 사용한다.

Column명은 DB의 Column명과 동일하게 사용한다.

1.4 Function (함수명)

1.4.1 공통함수

- 1) 프로젝트에서 공통으로 사용하는 Global Function

접두어 "g"에 "fn"을 붙인 후, 의미있는 단어의 조합으로 표시한다.

예] gfnIsNull(), gfnGetLoginInfo();

- 2) 공통버튼 함수 (조회/추가/삭제/저장)

공통 버튼 영역에 버튼을 클릭할 경우 각 화면의 접두어 "c"와 "fn"을 붙인 후, 의미 있는 단어의 조합으로 구성된 Function을 호출한다.

유형	의미 있는 단어	함수 명	비고
조회	Search	cfnSearch()	공통 조회 버튼
저장	Save	cfnSave	공통 저장 버튼
추가	Add	cfnAdd	공통 추가 버튼
삭제	Del	cfnDel	공통 삭제 버튼

표 1-4 공통 버튼 함수 명

- 3) 각 업무 화면에서 사용하는 Global Function

Biz 하위 폴더 약명에 "fn"을 붙인 후 의미 있는 단어의 조합으로 표기

예] bizBase 폴더 내 공통 함수 : bsfnGetBoardInfo();

1.4.2 일반 함수

Form Script 안에 선언되며, Form내에서만 사용 될 수 있다.

- 1) Event 함수

이벤트 발생 시 작동하는 함수 명은 개발 툴인 Nexacro Studio에서 자동으로 생성하는 component id + "_" + 이벤트 명으로 유지한다.

예] btnSearch_onclick(){...}, cboGubun_onchanged(){...}

- 2) 개발자 정의 함수

“fn” 으로 시작하고 함수의 기능을 표현하는 의미 있는 단어의 조합으로 표기한다.

함수명은 동사로 짓고, 여러 단어를 사용할 경우 단어의 배열은 [동사+명사]의 순서를 따른다.

메타에 명시되어 있는 단어를 이용하며, 가능한 원 용어를 사용하되 단어가 지나치게 긴 경우 약어를 사용한다.

아래에 기술된 공통 단순 기능(조회, 등록, 수정, 삭제, 검증 등)의 경우 동사를 아래의 용어를 포함해서 명명한다.

예] fnCheckValid(){...}

구분	Prefix	적용 예
개발자 정의 함수	fn[동사][명사]를 사용한다.	this.fnCheckValid = function(){...}

표 1-5 개발자 정의 함수 예시

1.5 Variable (변수)

Data Type	약어	적용 예시	비고
Number	n	nCnt	
String	s	sValue	
Boolean	b	bResult	
Object	obj	objGrid	
Array	arr	arrMessage	

표 1-6 DataType별 약어

1.5.1 글로벌 변수

Nexacro studio의 project Explorer상 Environment와 Application Variable, Dataset을 의미한다.

화면 내 변수와 구분하기 위해 Environment Variables는 접두어 “ev”를 붙여 명시하고, Application Variables는 접두어 “gv”를 붙여 명시한다. Application Dataset의 경우 글로벌 접두어

"g"와 Dataset Prefix "ds"를 붙여 "gds"를 붙여 명시한다.

예] Environment Variables : evTraceMode / Application Variables : gvUserId / Application Dataset : gdsMenu

1.5.2 화면 전역 변수

접두어 "fv" + 의미 있는 단어의 조합으로 표기한다.

예] this.fvBoardCd; this.fvBoardNm

주의 : 화면 내 전역변수는 해당 화면(form)내에서 Unique해야 하며, Form 변수 선언 영역에 this.fvBoardNm = ""; 과 같이 선언해야 한다.

1.5.3 로컬 변수

개발자 함수 내에서 사용하는 변수로, 데이터 타입 + 의미 있는 단어의 조합으로 사용한다.

예] var nCnt = 0; var sTitle = "로컬변수";

주의 : Script내 변수 선언 시, Var를 선언 하지 않으면 Global 변수로 생성되므로 반드시 var 를 선언하여 사용해야 한다. Var nCnt = 0;

2. 코딩 표준

2.1 기본 사항

- 1) 모든 내용은 들여쓰기를 적용하며 indentation은 4칸(nexacro 기본 tab size)로 적용
- 2) 구성요소 사이에 공백은 1칸을 기본으로 한다.
- 3) 한 구문에서 여러 대입식을 사용하지 않는다.
- 4) 쌍을 이루는 괄호는 반드시 줄을 맞춘다.

2.2 줄 바꿈

한 줄인 경우에도 {}로 묶는다.

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

예] Argument가 많아 한 줄에 쓸 수 없거나 가독성이 떨어지는 경우

```
this.fnOpen = function (sPrefix, sFormID, sInArgument, iWidth, iHeight,
                        sOpenStyle, iLeft, iTop)
{
    this.fnCall( );
}
```

예] 조건, Dataset Argument 등으로 사용할 String을 연결 할 경우(가독성이 떨어지는 경우)

```
this.fnClick = function (objBtn)
{
    var outDs = "";
    var params = " initMsgList=T"
                + " userId=69709"
                + " custNm=KT"
                + " rnsCnfmYn=Y"
                + " custPswd=ok";
}
```

예] 조건이 복잡하고 많아 가독성이 떨어지는 경우 (조건별 개행 및 줄 맞춤)

```

if ( (condition1 && condition2)
    || (condition3 && condition4)
    || (condition5 && condition6)
)
{
    statements;
}

```

2.3 빈 줄 삽입

빈 줄에 대한 규정은 없으나, 가독성을 위하여 블록 이후, 성격이 다른 경우, 기타 가독성을 위한 경우 빈 줄 삽입을 권장한다.

2.4 Block Statement

예] if문은

```

if (condition) {
    statements;
}else if (condition) {
    statements;
}else {
    statements;
}

```

예] for문/ while / switch 문

```

for (var i = 0; i < iCnt; i++){
    statements;
}

while (i > iCnt){
    statements;
}

```

```

}

switch (svcID)
{
    case "search":
        statements;
        break;
    default: break;
}

```

2.5 Comments (주석)

2.5.1 스크립트 표준 (화면 표준 주석)

모든 화면의 script의 주석, 기능별 함수의 위치, 공통 기능 함수 명 등의 표준화를 통해 개발 및 운영 시 효율성과 가독성을 최대화하여 개발 생산성을 극대화한다.

```

/**
 * DevPACK
 *
 * @FileName   sampleScript.xfdl
 *
 * @Creator    TOBESOFT
 *
 * @CreateDate 2020/11/24
 *
 * @Desction
 *
 * 소스 수정 이력
 *
 * Date          Modifier          Description
 *
 * 2020/11/24    TOBESOFT          최초생성
 *
 */

```

```

/*****

* FORM 변수 선언 영역

*****/

/*****

* FORM EVENT 영역(onload, onbeforeclose..)

*****/

this.form_onload = function(obj:nexacro.Form,e:nexacro.LoadEventInfo)
{
    this.gfnFormOnload(obj,e); //필수함수
};

/*****

* 공통함수영역 (cfnSearch : 조회 / cfnSave : 저장 / cfnAdd : 신규 / cfnDel : 삭제 )

*****/

/*****

* Transaction 서비스호출 처리 영역

*****/

/*****

* Callback 영역 (Transaction, popup, message..)

*****/

```

```

/*****

* 사용자 Function 영역

*****/

/*****

* 각 COMPONENT 별 EVENT 영역

*****/

```

2.5.2 함수 주석

- 1) Comments는 /* ... */ , // 둘 다 사용 가능하다
- 2) 주석의 종류는 class, param, return를 반드시 작성한다.
- 3) jsdoc 에서 지정한 주석 표준을 사용한다

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

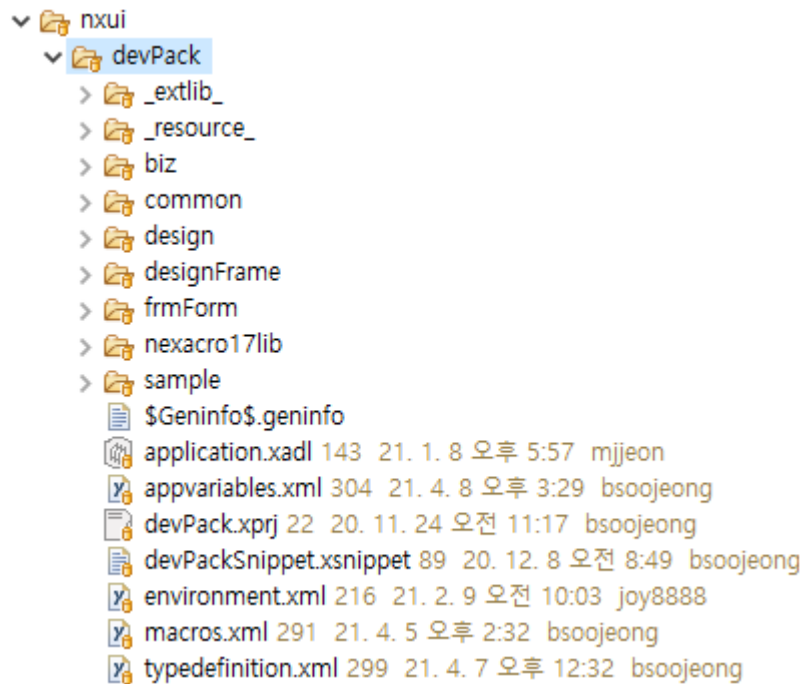
표 3-1 공통 함수 주석 표준

파트4. 개발 가이드

3. 프로젝트 구성

3.1 UI 프로젝트 폴더 구조

3.1.1 실제 넥사크로 소스



폴더명	PrefixID	설명	비고
extlib		Nexacro install module	
resource		theme, initvalue, images, font	
biz		업무화면 최상위 폴더	
Biz/auth	bizAuth	권한 관련 업무 화면	
Biz/base	bizBase	기준 정보 관련 업무 화면	
frmForm	frmForm	Frame관련 화면	
common	common	공통화면파일	

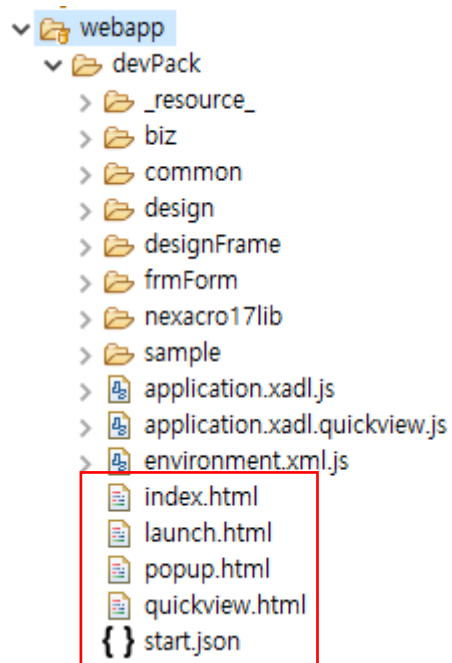
폴더명	PrefixID	설명	비고
sample	sample	샘플 화면	
design	design	디자인템플릿	
designFrame	designFrame	프레임관련 디자인 템플릿	
nexacro17lib		nexacro17 frameworklibrary	
DevPackLib	devPackLib	devPack 공통 Library	nexacro17lib₩component₩devPackLib

표 3-1 UI프로젝트 폴더 구조

3.1.2 Generate 소스

넥사크로 스튜디오에서 편집한 파일은 실행하기 위해 자바스크립트 파일로 변화하는 과정이 필요하다. 이 동작을 Generate라고 한다.

Generate된 소스는 넥사크로소스와 동일한 디렉토리로 생성되며, 추가적으로 몇가지 html파일이 생성된다.



파일	설명
index.html	Web Browser 배포 파일 Packing 시 생성됩니다. 별도 파라미터 없이 서버에 배포하고 해당 파일 URL 접근 시 실행합니다.

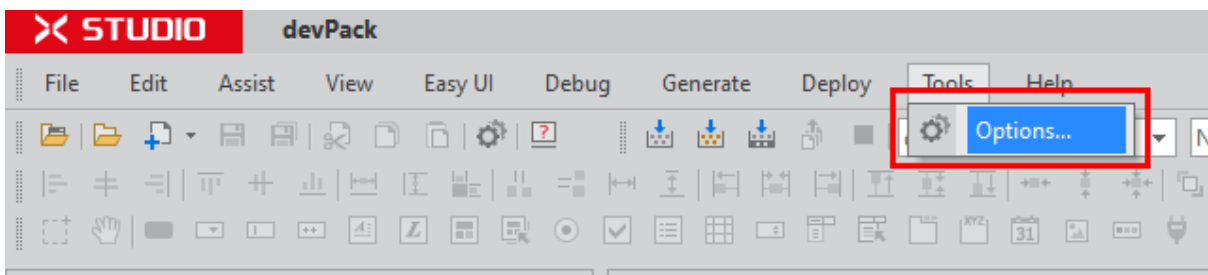
파일	설명
launch.html	넥사크로 스튜디오에서 [Launch] 실행 시 사용하는 파일입니다. Generate 시 생성되며 배포 시에는 사용하지 않습니다. screenid 를 파라미터로 받아서 실행합니다.
quickview.html	넥사크로 스튜디오에서 [QuickView] 실행 시 사용하는 파일입니다. Generate 시 생성되며 Web Browser 배포 파일 Packing 시에도 생성됩니다. screenid 와 formname 을 파라미터로 받아서 실행합니다. 특정한 화면을 실행해야 하는 경우 해당 파일을 활용할 수 있습니다. 예를 들어 컴포넌트 활용 가이드 에서 DataObject 관련 예제 실행 시 quickview.html 템플릿을 일부 변형해 사용하고 있습니다. http://demo.nexacro.com/developer_guide/17/guide.html?sa=sample_dataobject_01&ma=ca48b35c0d288342
popup.html	nexacro.open 메소드로 모달리스 윈도우 실행 시 사용하는 파일입니다. Generate 시 생성되며 Web Browser 배포 파일 Packing 시에도 생성됩니다.
run.html	iOS 운영체제 배포 파일 Packing 시 생성됩니다. Update Type 이 "Server"인 경우에는 Run.html 파일로 생성되며 "Update", "Local"인 경우에는 Run.zip 파일 내에 압축된 형태로 생성됩니다.

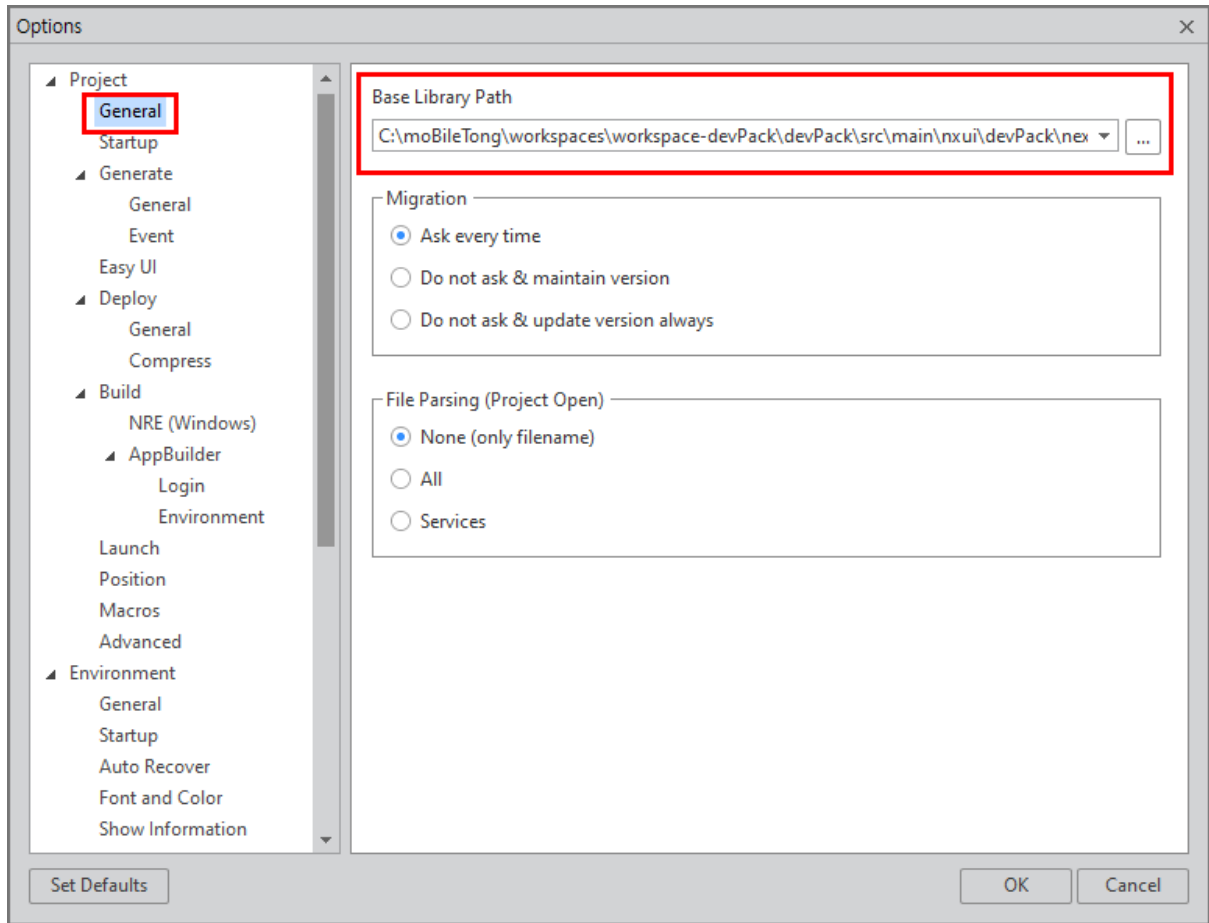
3.2 Nexacro Studio 설정

3.2.1 Base Library Path

프로젝트마다 사용하는 기본 라이브러리가 다를 경우 해당하는 경로를 지정한다.

Nexacro Studio 에서 [Tool] – [Options...] 을 클릭 후 나오는 Popup창에서 Base Library Path를 설정한다.





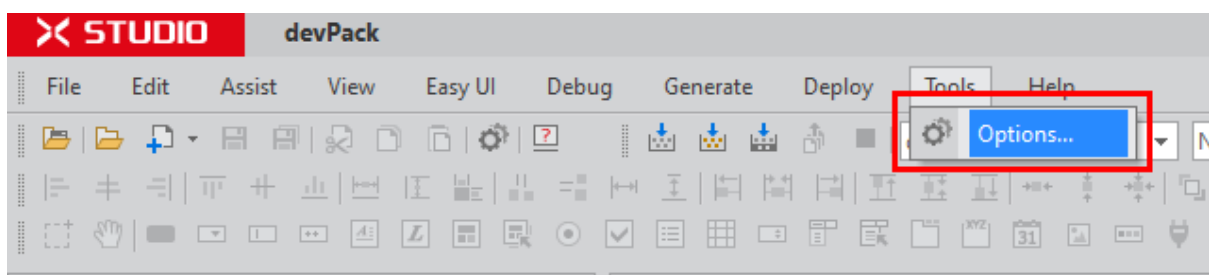
devPack Base Library Path (프로젝트마다 상이함) :

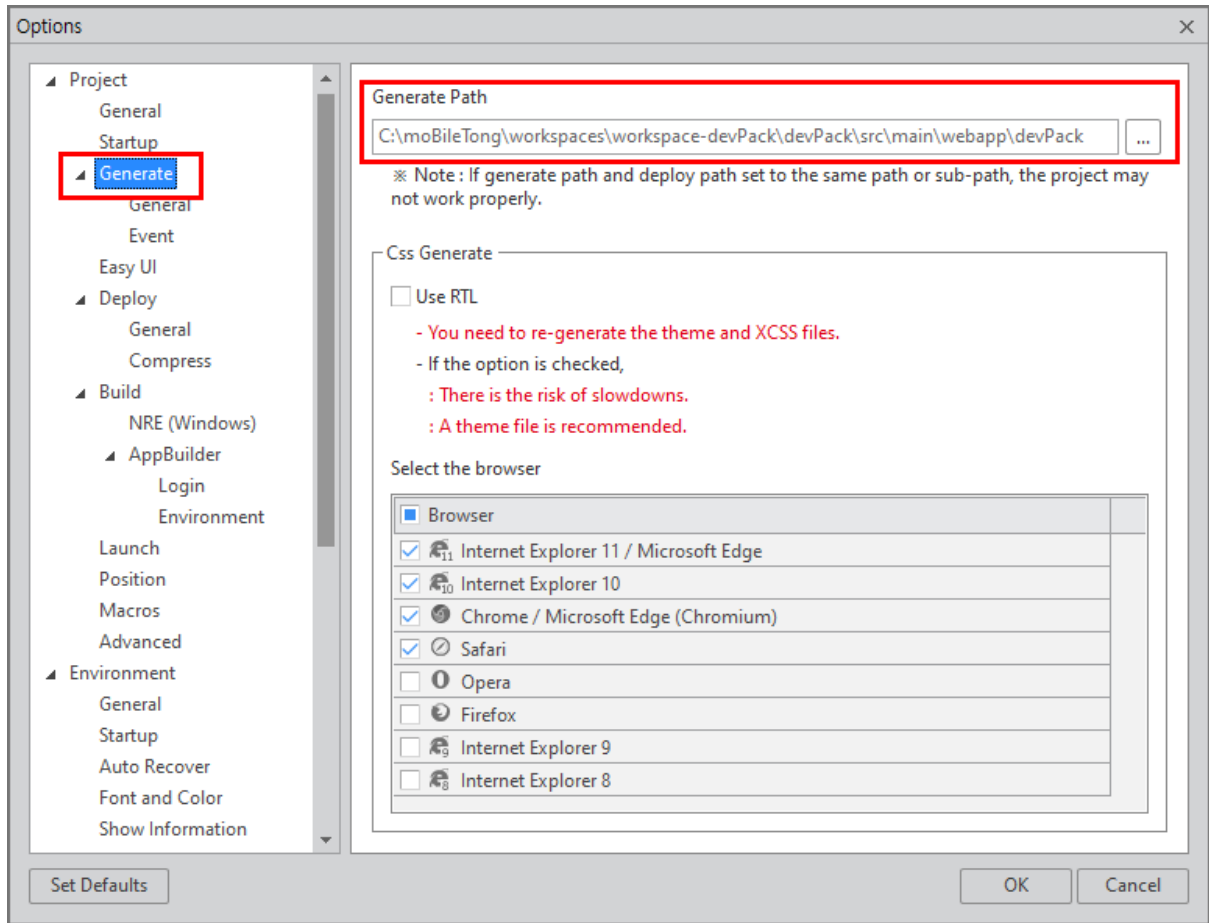
C:\moBileTong\workspaces\workspace-devPack\devPack\src\main\nxui\devPack\nexacro17lib

3.2.2 Generate Path

프로젝트 Build시 생성된 파일이 저장되는 경로를 지정한다. Generate 경로를 Deploy 경로와 동일한 경로나 하위 경로로 설정하면 프로젝트가 제대로 동작하지 않을 수 있다.

Nexacro Studio 에서 [Tool] – [Options...] 을 클릭 후 나오는 Popup창에서 Base Library Path를 설정한다.





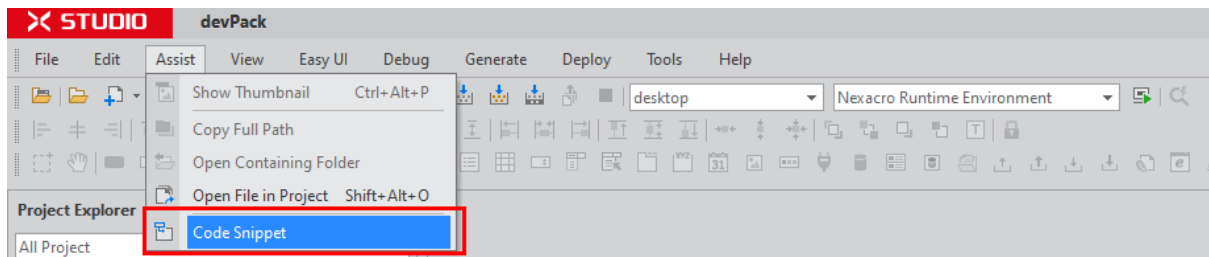
devPack Generate Path (프로젝트마다 상이함) :

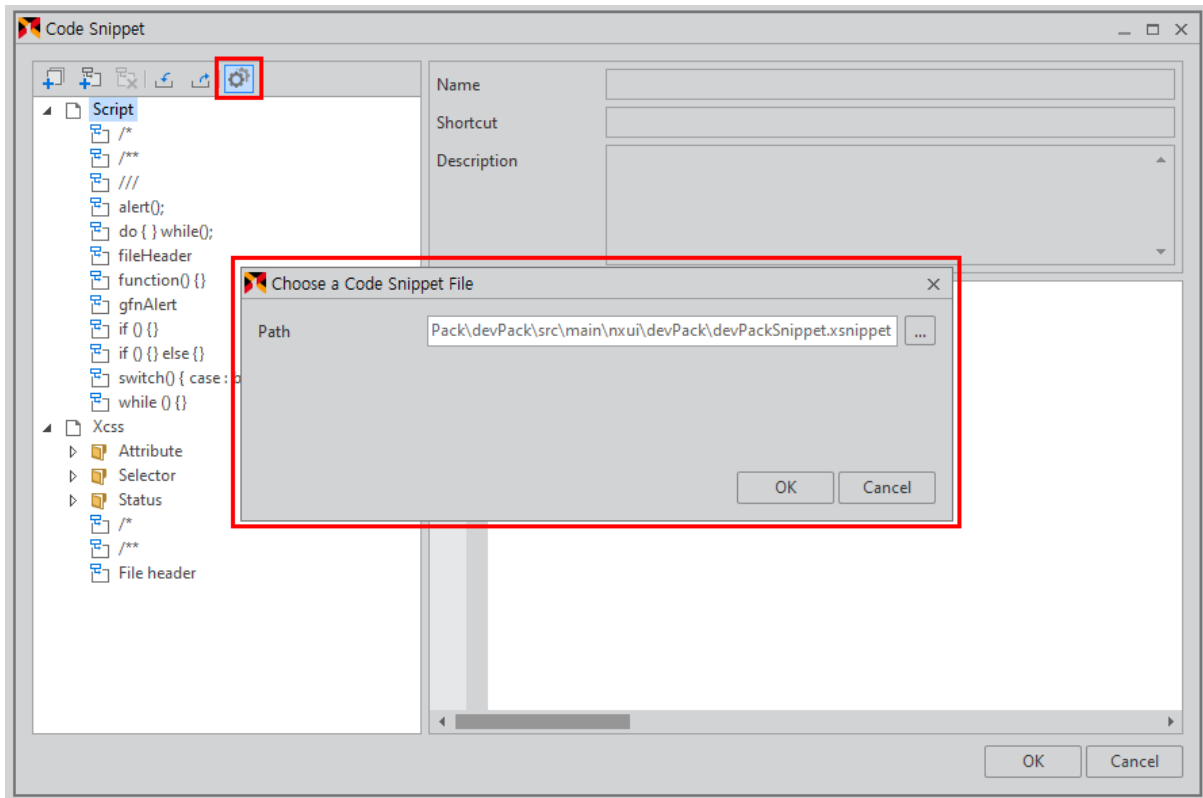
C:\moBileTong\workspaces\workspace-devPack\devPack\src\main\webapp\devPack

3.2.3 Code Snippet 설정

Code Snippet은 사용자가 미리 정의한 코드 조각을 의미한다. 자주 사용하는 코드를 미리 정의하고 파일 편집 시 가져다 사용할 수 있는 기능이다.

Nexacro Studio 에서 [Assist] – [Code Snippet] 을 클릭 후 나오는 Popup창에서 제공된 스니펫 파일 경로를 설정 할 수 있다.





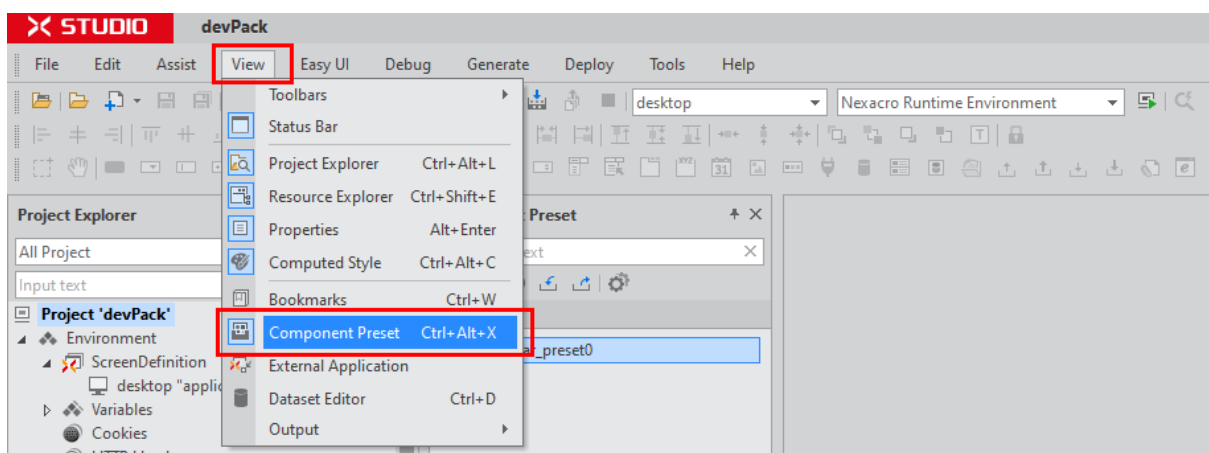
devPack Snippet 파일 경로 :

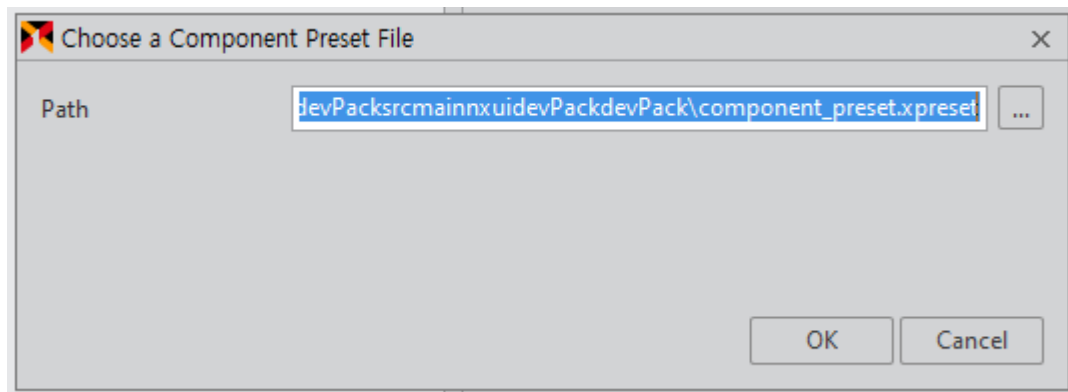
C:\#moBileTong\workspaces\workspace-devPack\devPack\src\main\nxui\devPack\devPackSnippet.xsnippet

3.2.4 Component Preset 폴더 설정

컴포넌트 프리셋은 반복적으로 지정해주어야 하는 컴포넌트의 속성, 이벤트 등의 정보를 별도의 프리셋 형태로 관리하고 재사용 할 수 있도록 지원하는 기능이다.

Nexacro Studio 에서 [View] – [Component Preset] 을 클릭 후 나오는 Component Preset 패널에서 등록된 컴포넌트를 관리 할 수 있다.





devPack Component Preset File Path :

*C:\Users\bsoojeong\Documents\nexacro\17.1\settings\ProjectConfigure\CmoBileTongworks
pacesworkspace-devPackdevPacksrmainnxuidevPackdevPack\component_preset.xpreset*

3.2.5 deploy 설정

프로젝트 Build시 생성된 파일이 저장되는 경로를 지정한다. Generate 경로를 Deploy 경로와 동일한 경로나 하위 경로로 설정하면 프로젝트가 제대로 동작하지 않을 수 있다.

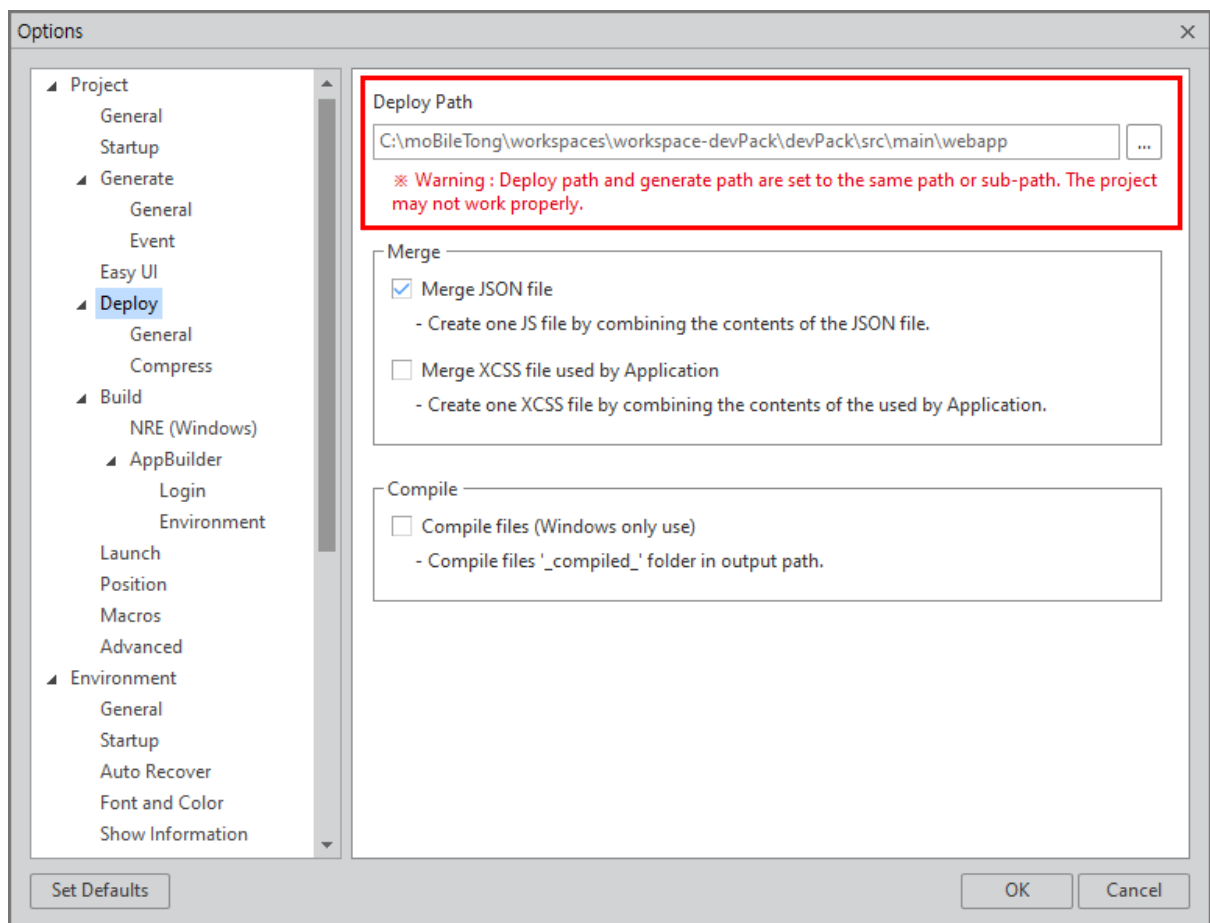
Nexacro Studio 에서 [Tool] - [Options...] 을 클릭 후 나오는 Popup창에서 [Deploy] - [General] 에서 Deploy path를 설정 한다.

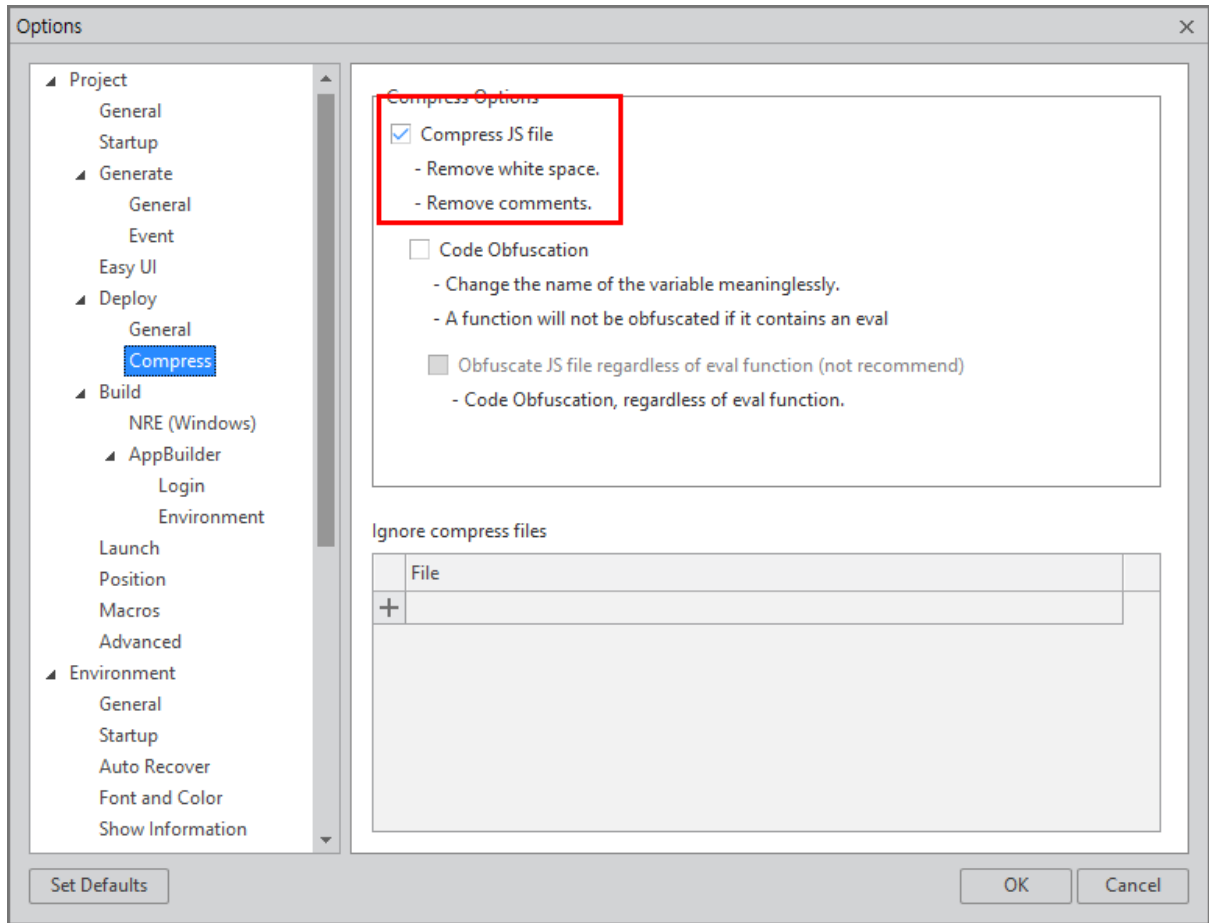
Deploy의 경우 Generate와 다르게 Merge/Compress옵션을 설정할 수 있다. 각 프로젝트 상황에 맞게 Merge/Compress 옵션을 지정할 수 있다..

[Deploy] - [General] 에서 Merge 옵션을 선택 하고 [Deploy] - [Compress] 에서 Compress옵션을 설정 한다.

Deploy Path : C:\moBileTong\workspaces\workspace-devPack\devPack\src\main\webapp

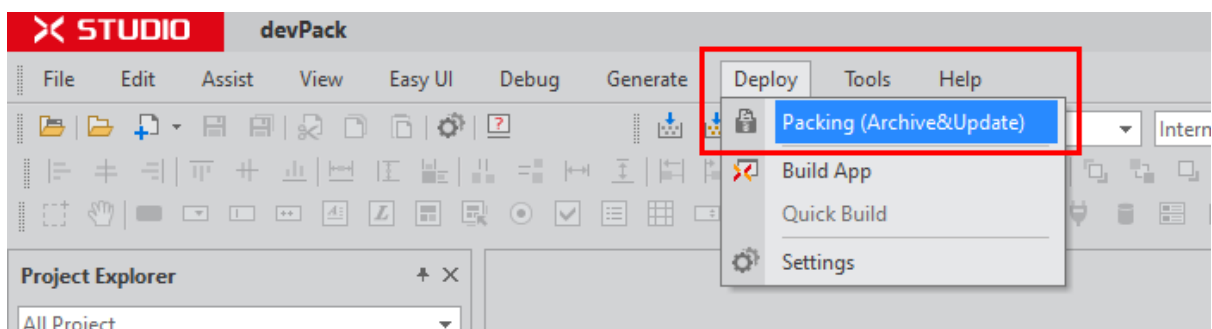
(추후 Deploy 옵션을 선택 할 때 Sub Derectory설정을 필수로 해야 Deploy를 수행할 수 있으므로 Webapp까지만 설정한다 단, Deploy Target 이 여러 가지 일 경우 각 환경에 맞게 세팅 한다.)

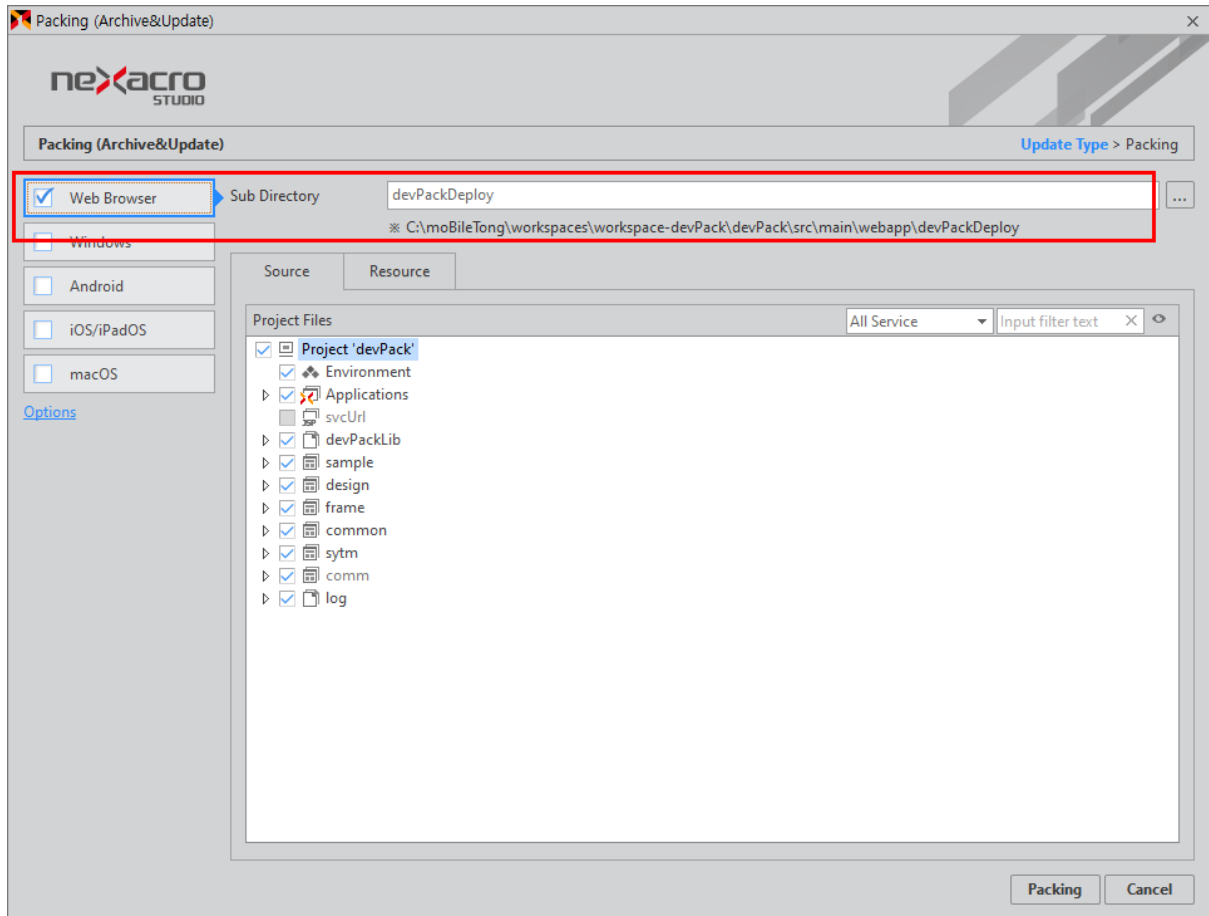




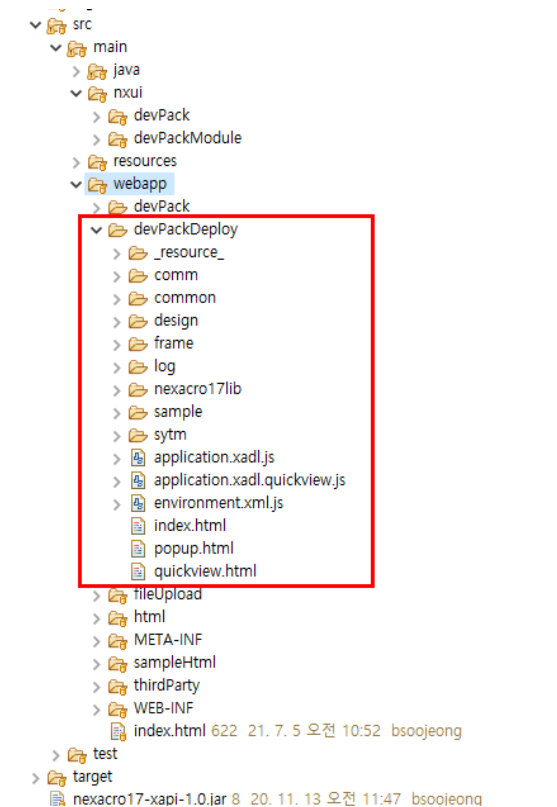
실제 Deploy는 [Deploy] - [Packing] 메뉴에서 실행한다.

메뉴 선택 시 나오는 popup창에서 Deploy Target (어떤 환경으로 배포할 지) 선택 후 각각에 맞는 Sub Directory 설정 후 Deploy할 소스와 리소스를 정확히 체크 후 Deploy를 진행한다.





Deploy 결과물은 아래와 같다. (현재 프로젝트는 Web Browser용 Deploy만 진행)



3.3 Environment Variables 와 Application Variables

id	initval
evMessagePopup	true
evQuickView	Y
evSytmFlagCd	00000001

id	initval	설명	비고
evMessagePopup	true	메세지 팝업 사용 유무	
evSytmFlgCd	Y	시스템 코드	
evQuikView	Y	QuickView 실행 여부	

표 3-2 Environment Variables

Application variables는 application 스크립트 영역에 선언되어 있다.

```

16
17 this.gvRunMode = ""; // 실행환경 S : Studio, L : local, D : 개발, R : 운영
18
19 /* 각 프레임에 해당하는 객체 참조 변수 */
20 this.gvVfrs;
21 this.gvFrmLogin;
22 this.gvFrmTop ;
23 this.gvHfrs;
24 this.gvFrmLeft;
25 this.gvVfrsWork;
26 this.gvFrmMdi;
27 this.gvFrmMain;
28 this.gvFrsWork;
29
30 /* 프레임 관련 변수*/
31 this.LOGIN_FORM_PATH = "frmForm::frmLogin.xfdl";
32 this.TOP_FORM_PATH = "frmForm::frmTop.xfdl";
33 this.LEFT_FORM_PATH = "frmForm::frmLeft.xfdl";
34 this.WORK_FORM_PATH = "frmForm::frmWork.xfdl";
35 this.MDI_FORM_PATH = "frmForm::frmMdi.xfdl";
36 this.MAIN_FORM_PATH = "frmForm::frmMain.xfdl";
37
38 this.gvOpenMaxFrame = 10; // 열리는 프레임 최대 갯수
39 this.gvMdiFramePos = "top"; // MDI Frame 위치 설정(top || bottom)
40 this.gvFrameStat = "login"; // 프레임상태(login,main,sub)
41 this.gvCloseCheck; // 화면 닫을때 체크할지 여부
42 this.gvIsComBtnUse = true; // 공통 버튼 사용유무
43 this.gvTitlebarHeight = 0; // 차일드프레임 타이틀바 높이
44
45 /* grid 관련 */
46 this.gvUseGridContextMenu = true; // 그리드 Context Menu 사용 여부
47

```

id	initval	설명	비고
gvVfrs		프레임셋 관련 변수	
gvFrmLogin		프레임셋 관련 변수	
gvFrmTop		프레임셋 관련 변수	
gvHfrs		프레임셋 관련 변수	
gvFrmLeft		프레임셋 관련 변수	
gvVfrsWork		프레임셋 관련 변수	
gvFrmMdi		프레임셋 관련 변수	
gvFrmMain		프레임셋 관련 변수	
gvFrsWork		프레임셋 관련 변수	
LOGIN_FORM_PATH	frmForm::frmLogin.xfdl	프레임 폼 연결 URL	
TOP_FORM_PATH	frmForm::frmTop.xfdl	프레임 폼 연결 URL	
LEFT_FORM_PATH	frmForm::frmLeft.xfdl	프레임 폼 연결 URL	
WORK_FORM_PATH	frmForm::frmWork.xfdl	프레임 폼 연결 URL	
MDI_FORM_PATH	frmForm::frmMdi.xfdl	프레임 폼 연결 URL	
MAIN_FORM_PATH	frmForm::frmMain.xfdl	프레임 폼 연결 URL	
gvOpenMaxFrame	10	열리는 프레임 최대 갯수	
gvMdiFramePos	Top	MDI Frame 위치 설정(top bottom)	
gvFrameStat	Login	프레임상태(login,main,sub)	
gvCloseCheck	N	화면 닫을때 체크할지 여부	
gvIsComBtnUse	True	공통 버튼 사용유무	

id	initval	설명	비고
gvTitlebarHeight	0	차일드프레임 타이틀바 높이	
gvUseGridContextMenu	Y	그리드 Context Menu 사용 여부	

표 3-3 Application Variables

3.4 Global Dataset

No	id	type	size	prop	sumtext	datapath	description
1	MSGE_CD	STRING	256				
2	KORN_MSGE	STRING	256				
3	MSGE_FLAG_CD	STRING	3				
4	msgTitle	STRING	256				

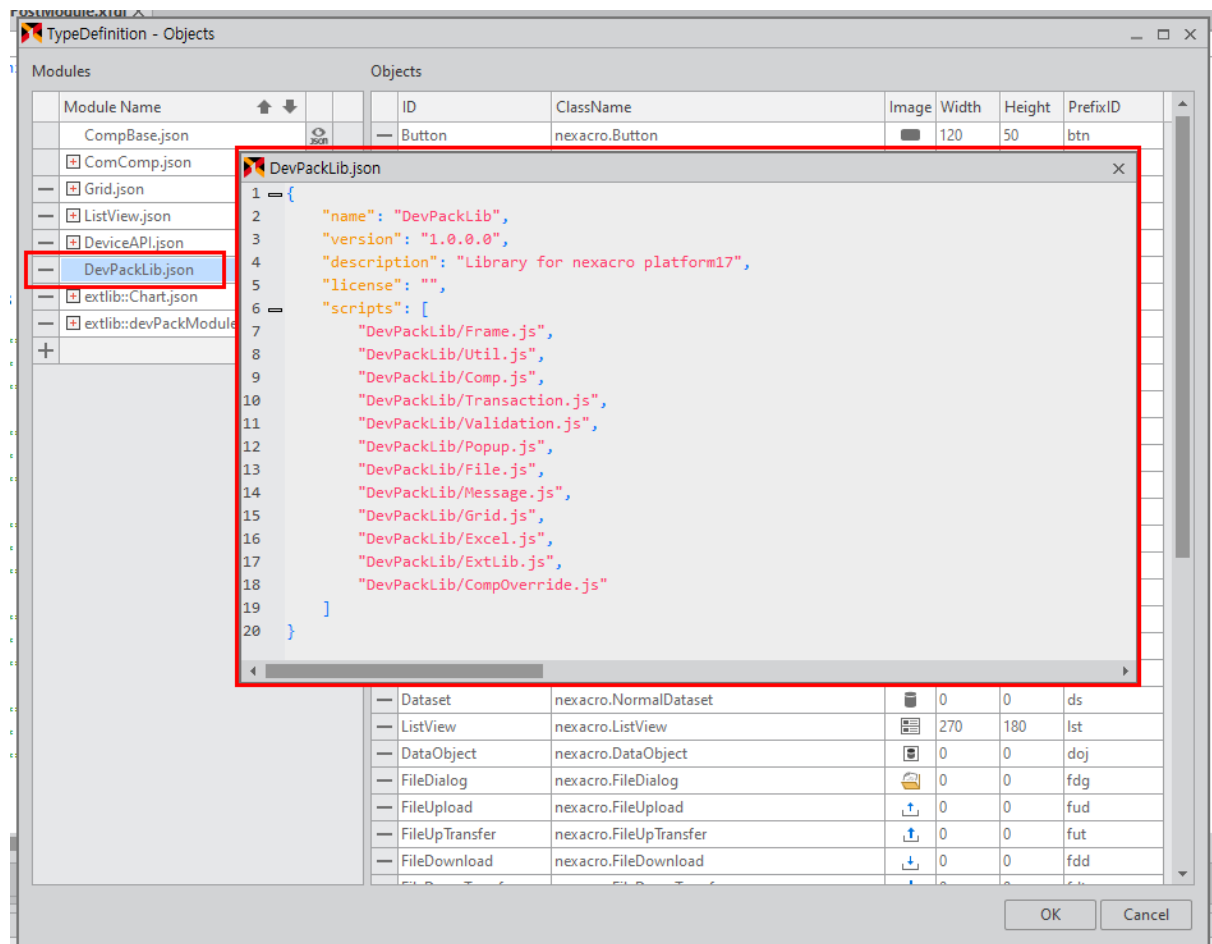
No	MSGE_CD	KORN_MSGE	MSGE_FLAG_CD	msgTitle
1	msg.server.error	서버 오류입니다.\n관리...	ERR	에러
2	msg.server.error.msg	서버에서 다음과 같은 ...	ERR	에러
3	msg.session.timeout	세션이 종료되었습니다...	WAN	경고
4	msg.login.url.error	정상적인 경로로 접속하...	ERR	에러
5	msg.login.error	해당하는 사용자 정보가...	WAN	경고
6	msg.call.nofile	해당하는 메뉴에 Progra...	WAN	경고
7	msg.nomenu	해당 Menu가 존재하지 ...	WAN	경고
8	msg.err.mdicount.max	화면은 {0}개까지만 실...	WAN	경고
9	confirm.logout	로그아웃 하시겠습니까?	CFN	확인
10	confirm.before.movepa...	변경된 데이터가 있습니...	CFN	확인
11	confirm.before.remove...	변경된 데이터가 있습니...	CFN	확인
12	confirm.before.reopen	변경된 데이터가 있습니...	CFN	확인
13	confirm.before.search	검색을 진행하면 변경된...	CFN	확인

id	설명	비고
gdsMessage	메세지 정보	
gdsOpenMenu	열린 메뉴 정보	
gdsMenu	메뉴	
gddGridPopupMenu	그리드 팝업 메뉴	
gdsCmmnBtn	공통버튼	

id	설명	비고
gdsCommCode	공통코드	
gdsUser	사용자 정보	
gdsLog	로그쌓기용	
gdsMyMenu	마이메뉴	

표 3-4 Global dataset

3.5 공통 Library



NO	파일명	파일의 기능	비고
1	Frame.js	프레임 관련 공통 함수	

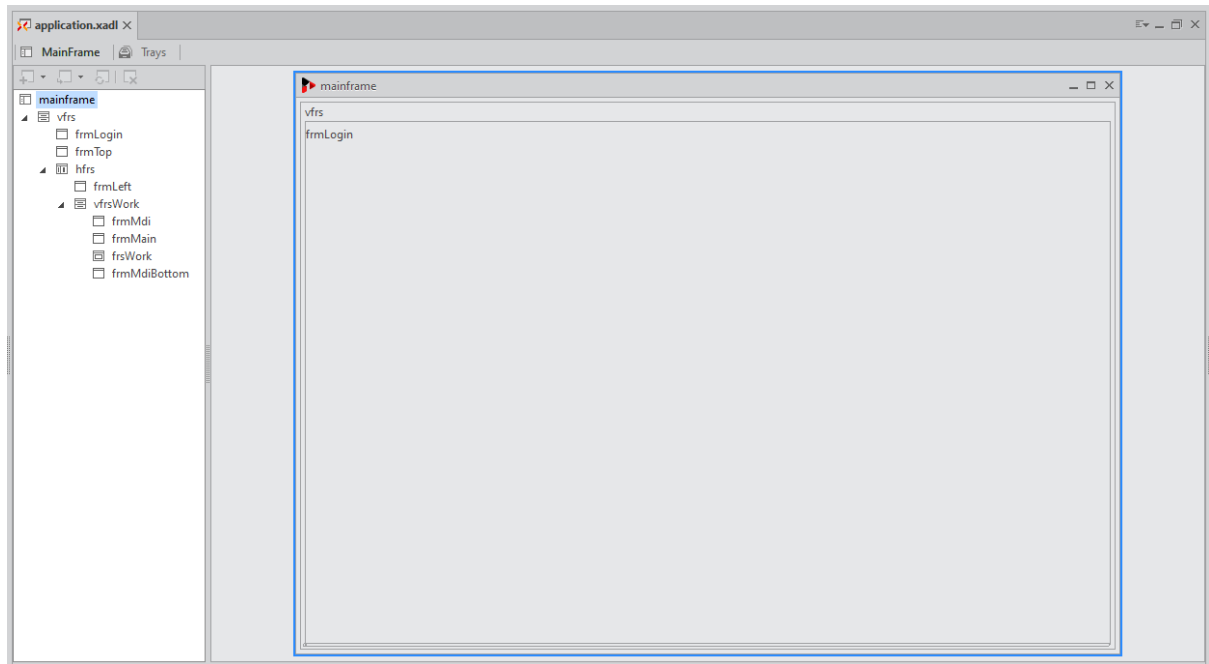
NO	파일명	파일의 기능	비고
2	Util.js	Date, String 관련 공통 함수	
3	Transaction.js	Service 관련 공통 함수	
4	Validation.js	Vaidation 관련 공통 함수	
5	Popup.js	팝업 관련 공통 함수	
6	File.js	File 업/다운로드 관련 공통 함수	
7	Message.js	메세지 처리 관련 공통 함수	
8	Grid.js	그리드 관련 공통 함수	
9	Excel.js	엑셀 import/export 관련 공통 함수	
10	ExtLib.js	기타 업무 베이스 공통 함수	

3.6 Frame 구조

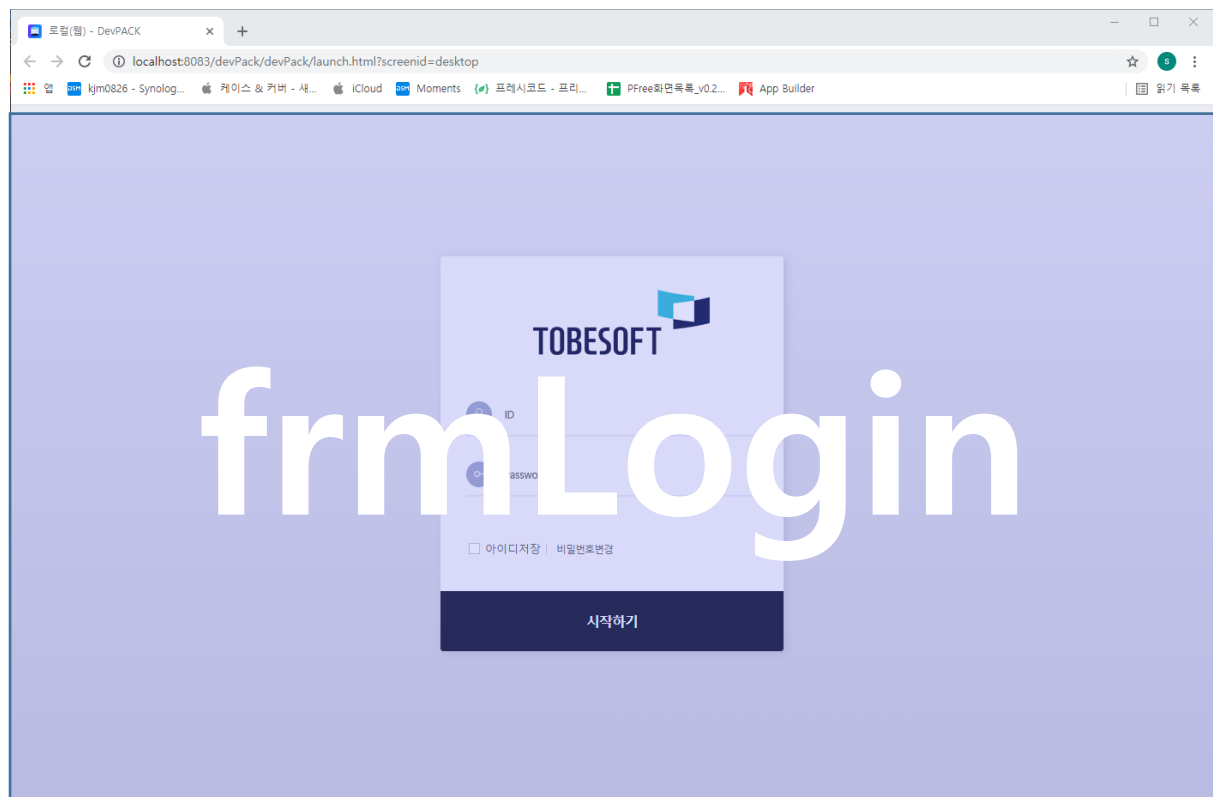
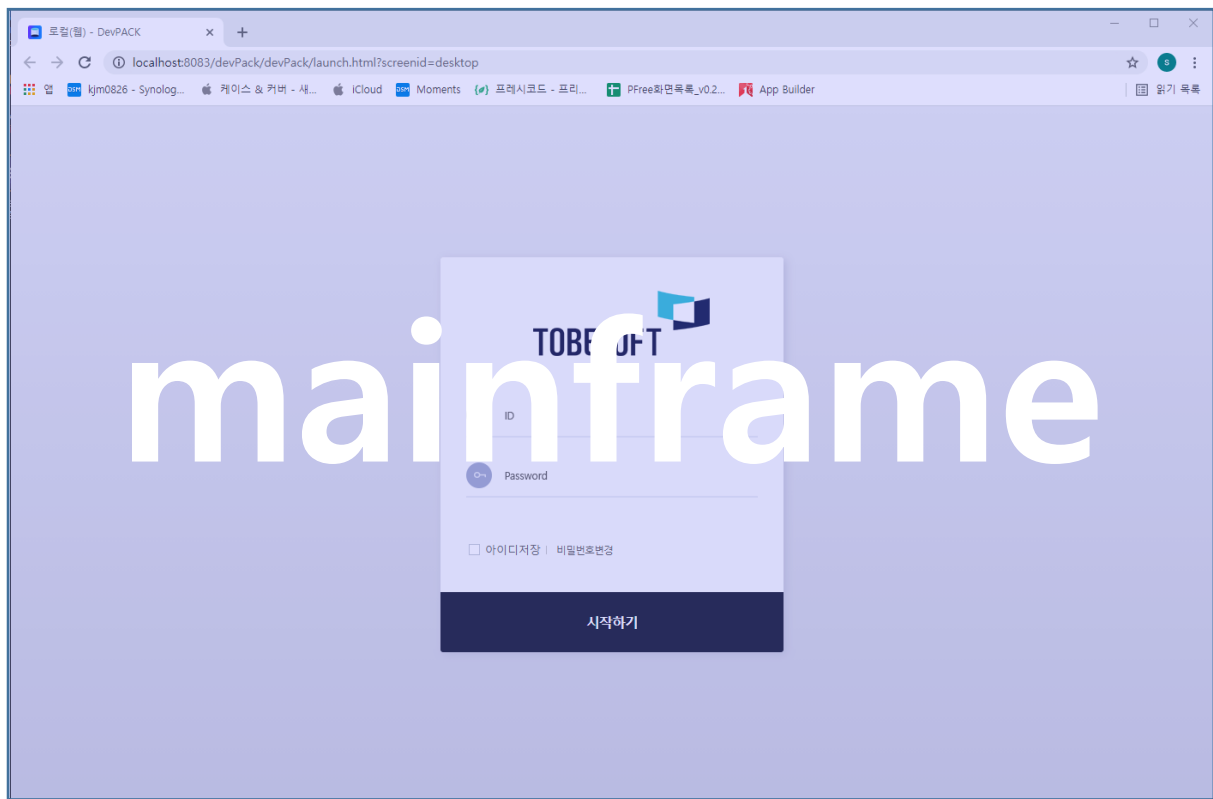
독립적인 창 구조를 가지고 동작하는 애플리케이션을 구성할 때 Frame 구조를 가진다. Mainframe은 그 이름처럼 frame구조를 이루는 기본 오브젝트로 한 애플리케이션당 하나만 존재할 수 있다.

웹브라우저에서 실행하는 경우에는 웹브라우저가 Frame의 역할을 담당한다. 콘텐츠의 로딩 상태나 화면의 크기, 타이틀 바 설정을 웹브라우저에서 처리한다. 웹브라우저의 특성에 따라 화면에 보이는 모습은 조금씩 다를 수 있다.

Frame 형태에 따라 Div(Division) 컴포넌트처럼 Container 컴포넌트를 사용하는 것인지 구분이 되지 않을 수 있다. Frames은 전체 애플리케이션의 구조를 지정하는 것이라면 Container 컴포넌트는 각 콘텐츠의 구조를 설정한다. Container 컴포넌트는 스크립트로 그 구조를 변경할 수 있지만, Frame 구조는 쉽게 변경하기 어렵다. Frame 구조는 제공하는 콘텐츠의 일관성을 유지하는 역할을 하고 있어 구조를 변경하지 않는 것이 좋다.



id	설명	비고
mainframe	DevPack Application의 기본 Frame	
vfrs	Vertical-Frameset	
frmLogin	Login Frame	
frmTop	Top Frame	
Hfrs	Horizon Frameset	
frmLeft	Left Frame	
vfrsWork	Vertical Frameset	
frmMdi	Mdi Framd	
frmMain	Main Frame	
frsWork	Work Frameset (실제 업무 화면 childframe을 가질 frameset)	
frmMdiBottom	Bottom Frame	



4. 기능별 개발 방법

4.1 Transaction

서버와 통신은 해당 프로젝트에 맞게 *Environmetn*의 *Httpretry*, *httptimeout*값을 조정해야 한다. 보통의 경우 *httpretry=0* 으로 설정하여 오류가 발생했을 때 해당 서비스를 재 호출하지 않도록 하며, *httptimeout*의 값은 WAS의 *timeout* 시간을 확인하고 네트워크 통신 시간을 감안하여 WAS *timeout +2* 초 정도의 시간으로 설정한다.

서버와 데이터를 송/수신 시 사용하는 공통 함수로 서비스 호출 URL, input/output Dataset, 콜백 함수, 통신 방식 등을 설정 할 수 있다.

```
/**
 * @class 서비스 호출 공통함수 <br>
 *
 * Dataset의 값을 갱신하기 위한 서비스를 호출하고, 트랜잭션이 완료되면 콜백함수를 수행하는 함수
 *
 * @param {String} strSvcId - 서비스 ID
 *
 * @param {String} strSvcUrl - 서비스 호출 URL
 *
 * @param {String} [inData] - input Dataset list("입력ID=DataSet ID" 형식으로 설정하며
빈칸으로 구분)
 *
 * @param {String} [outData] - output Dataset list("DataSet ID=출력ID" 형식으로 설정하며
빈칸으로 구분)
 *
 * @param {String} [strArg] - 서비스 호출시 Argument
 *
 * @param {String} [callBackFnc] - 콜백 함수명
 *
 * @param {Boolean} [isAsync] - 비동기통신 여부
 *
 * @return N/A
 *
 * @example
 * var strSvcUrl = "transactionSaveTest.do";
 * var inData = "dsList=dsList:U";
 * var outData = "dsList=dsList";
```

```

* var strArg    = "";
* this.gfnTransaction("save", strSvcUrl, inData, outData, strArg, "fnCallback", true);
*/

```

서버에서 데이터 수신 시, 공통 Callback 함수인 gfnCallback 함수에서 서버 에러에 대한 메시지 처리 후 gfnTransaction 호출 시 지정한 업무화면의 콜백함수를 호출한다. 업무화면의 콜백함수에서는 오류 메시지를 처리하지 말고 error code – 에러코드 (정상 0, 에러 음수값)에 따라 업무화면에서 처리할 스크립트를 기술한다.

Application 성능 향상을 위해 통신방식은 Async(비동기) 통신을 사용하고 응답 후 처리는 콜백 함수에 기술해야한다. 업무화면에서 Sync(동기) 통신 사용시에는 반드시 공통 개발자와 상의 후 사용해야 하며, 여러 개의 통신을 여러 번 호출하는 것보다 1개의 통신에 여러 개의 input / output dataset을 보낸 후 서버에서 일괄 처리하면 application 성능이 향상 될 수 있다.

```

/**
* @description Transaction CallBack 함수(선택)
*/
this.fnCallback = function(svcID,errorCode,errorMsg)
{
    // 에러 시 화면 처리 내역

    if(errorCode != 0)
    {
        return;
    }

    switch(svcID)
    {

```

```

        case "search":
            // trace(this.dsList.saveXML());
            break;

        case "save":
            // 저장 되었습니다.

            this.gfnAlert("msg.save.success");
            break;
    }
};

```

조회 시 조회 조건이나 저장 시 입력 데이터를 서버로 보내는 input/output data는 Argument보다는 Dataset을 이용하여 처리하며, Component에 해당 Dataset을 bind 하여 사용하면 스크립트를 최소화 할 수 있다.

입력ID = Dataset ID:추가옵션(U/A/N)

추가 옵션을 통해 보내는 데이터의 양을 제어 할 수 있다.

:U 는 갱신된 정보

:A 는 모든 정보

:N 은 현재 삭제한 데이터를 제외한 정보를 서버로 송신한다.

Transaction 시 전송되는 데이터의 형태는 로컬(Studio, 웹)에서 실행 시에는 XML 형식으로 통신하여 통신 시 패킷 정보를 쉽게 확인 할 수 있도록 하고, 운영 실행 시에는 SSV형식으로 통신하여 통신 패킷의 크기를 줄여 성능을 향상 시킨다.

4.2 Grid 기능

정렬, No data text, 데이터 복사/붙여 넣기, context menu (컬럼 숨기기/보이기 , 열 고정, 행 고정, 데이터 필터, 데이터 찾기, 초기화) , 유저 헤더 그리드 기능이 제공 된다.

그리드에 아무것도 설정하지 않을 시, 정렬, No data text, 데이터 복사/붙여 넣기 기능을 사용 할

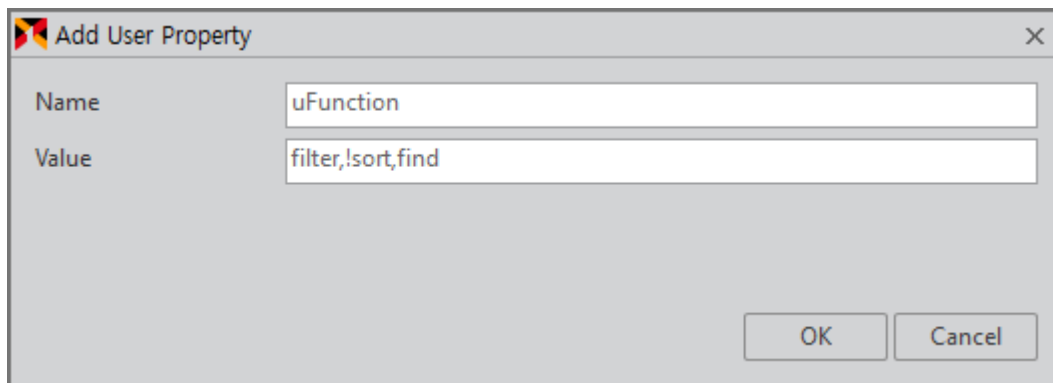
수 있다.

기본 기능 외 다른 기능을 추가 할 경우 그리드에 user properties를 추가하여 사용한다.

추가 방법 : 그리드를 선택 하고 프로퍼티창에서 마우스 우 클릭 - [add properties] 를 선택하여 유저 프로퍼티를 추가 후 사용할 기능을 콤마(,)로 구분하여 나열한다.

예] user property name : uFunction

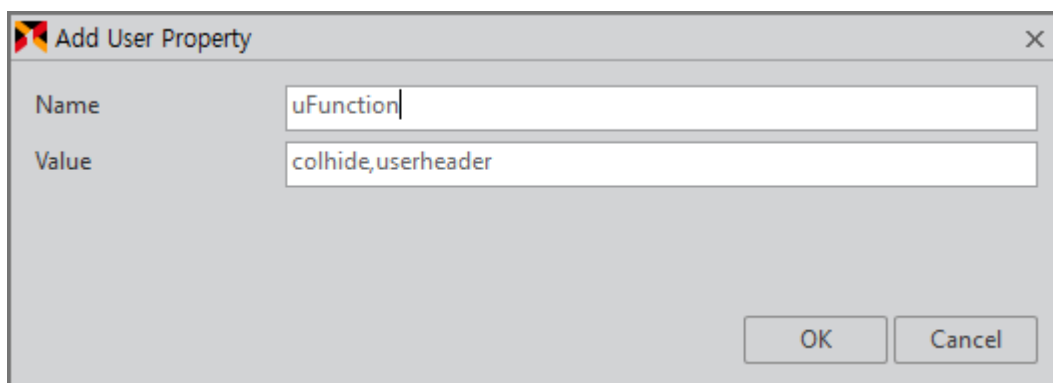
User property value : !sort,colhide

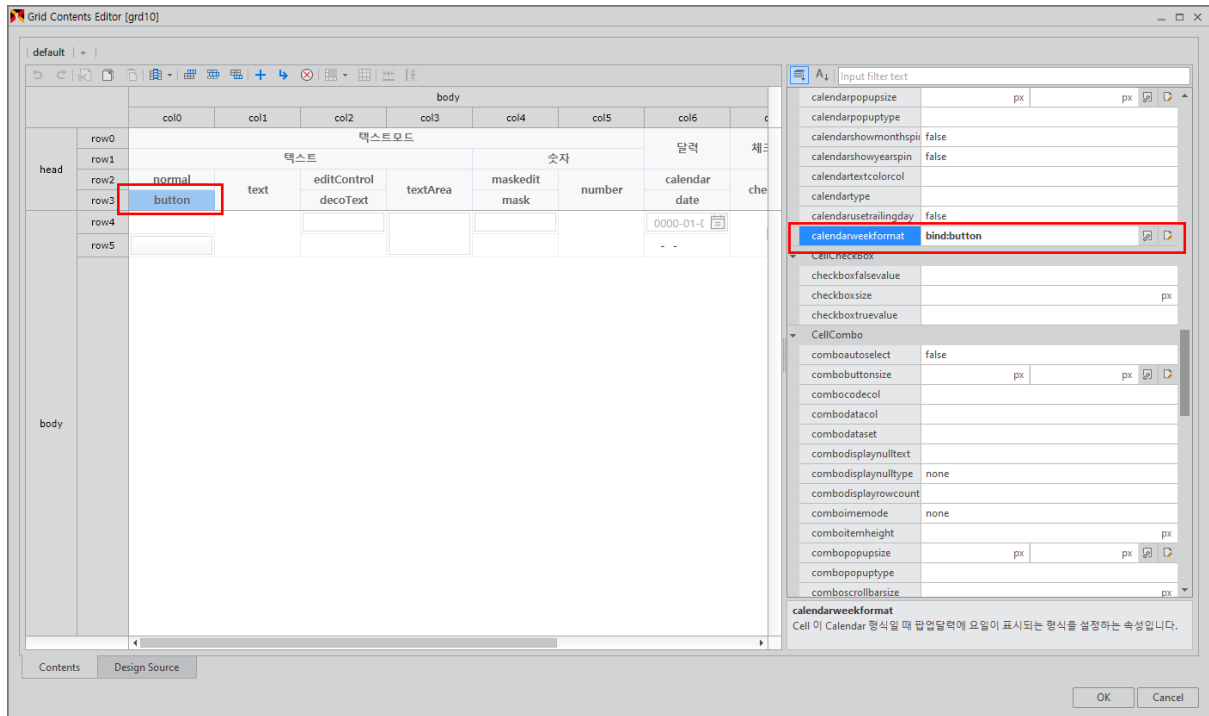
A screenshot of a dialog box titled "Add User Property". It has a close button (X) in the top right corner. The dialog contains two text input fields. The first field is labeled "Name" and contains the text "uFunction". The second field is labeled "Value" and contains the text "filter,!sort,find". At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

기본 기능 중 사용하지 않을 기능 앞엔 느낌표(!)를 붙여 명시한다.

예] !sort,!cellcopypaste

그리드 헤더와 로우가 매칭되지 않는 복잡한 그리드의 경우 userheader 속성을 사용 하여 필터, 소트 등을 손쉽게 이용 할 수 있다. 단 각 헤더정보에 기능을 이용할 바인딩 정보가 매칭되어야 하고 현재 헤더 셀 프로퍼티 [calendarweekformat] 속성에 바인딩 될 컬럼이 매칭 되어 있다.

A screenshot of a dialog box titled "Add User Property". It has a close button (X) in the top right corner. The dialog contains two text input fields. The first field is labeled "Name" and contains the text "uFunction". The second field is labeled "Value" and contains the text "colhide,userheader". At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".



소트, No data text, 데이터 복사/붙여 넣기 기능을 제외한 기능은 마우스 우클릭시 나오는 팝업 메뉴(Context menu)를 띄워 제공한다

2) 열고정, 그리드설정초기화

normal	button	calendar	check	combo	comboText	date	editControl	number	text
서울특별시	서울특별시	19890810	1	next	next	19890810	서울특별시	999999999	서울특별시
경기도	경기도	19890810	0	all	all	19890810	경기도	888888888	경기도
서울특별시	서울특별시	19890812	0	prev	prev	19890912	서울특별시	444444444	서울특별시
경기도	경기도	19890810	0	next	next	19890810	경기도	555555555	경기도
경기도	경기도	19890810	0	all	all	19890826	경기도	111111111	경기도
서울특별시	서울특별시	19890912	1	prev	prev	19890912	서울특별시	2	서울특별시
TEST	TEST	19890810	1	next	next	19890810	TEST	222222222	TEST
test	test	19890826	1	all	all	19890826	test	2	test

열고정 열: 열고정 해제, 열고정 설정 초기화

• 사용자 리스트

사용자 ID	생년월일	사용자 이름	부서	사용자구분	직원구분	직종	직급	재직상태	직책
nexter	1900-01-01	NEXTER	A-1-1팀	직원	정규직	시스템분석/설계/PM	부장	재직	팀원
test	1901-01-01	TESTER	B-1-1팀	직원	정규직	시스템/응용프로그래머	과장	재직	팀원
hsji	0000-01-01	지루박	B-1-1팀				과장	재직	
test1	1992-01-01	test							
test111	1992-01-01	테스트	A본부		정규직	경영/회계/재무	사원	재직	팀원

그리드 필터, 그리드 필터 해제, 데이터 찾기

4.3 Excel Export / import

각 Export/import별 공통함수를 사용하여 엑셀 export/ import를 실행한다.

우선적으로 제니(xeni)가 서버에 적용되어 있어야 실행 가능하다.

샘플 화면 : sample:: sampleExcel.xfdl

4.3.1 Export

```
//엑셀 익스포트

/**
 * @class excel export <br>
 * @param {Object|Array} objGrid - Grid Object
 * @param {String|Array} [sSheetName] - sheet name
 * @param {String} [sFileName] - file name
 * @param {String} [sExportType] - export 프로그램 Type
 * @param {String} [strRemoveCol] - Grid에서 export 시 제거할 Column 예) "1,2"
 * @param {String} [strTitle] - Grid에 추가할 타이틀
 * @param {Boolean} [bWordwrap] - 엑셀 export시 wordwrap 여부(default=true)
 * @param {String} [strUnit] - Grid에 상단 좌측에 단위 표시
 * @return N/A
 * @example
 * this.gfnExcelExport(this.grdMessage, "메시지", "Message",
this.cboExportType.value, "0,1,2", this.edtTitle.value);
 * this.gfnExcelExport([this.grdSuppress, this.grdMessage00], ["sheet!A1",
"sheet!A"+nCnt], "1개 sheet Export", null, ["0,1", "0,1,2,3"], ["메시지 목록",
"과일별 색상 목록"]);
 * this.gfnExcelExport([this.grdMessage00, this.grdSuppress],
["메시지", "suppress"], "2개 sheet Export");
 */
```

4.3.2 import

```
//엑셀 임포트
```

```

/**
 * @class excel import 할 엑셀에 칼럼명이 없을 경우 데이터셋의 칼럼 순번 기준으로 import
<br>
 * @param {String|Array} sDataset - dataset
 * @param {String|Array} [sSheet] - sheet name
 * @param {String|Array} [sBody] - body 영역지정
 * @param {String} [sCallback] - callback 함수
 * @param {String} [sImportId] - import id(callback호출시 필수)
 * @param {String} [sImportType] - import 프로그램 Type(defalut-EXCEL97)
 * @param {String|Array} [sHead] - head 영역지정(다중 sheet 업로드 시에는 필요)
 * @return N/A
 * @example
 * this.gfnExcelImport("dsMsg","메시지","A2","fnImportCallback","ExcelImport");           // dsMsg
칼럼 순번대로 엑셀의 A2 영역부터 import
 * this.gfnExcelImport(["dsTitle","dsMsg"], ["sheet","sheet"], ["A2:D2","A13"], "fnImportCallback",
"ExcelImportAll");
 */

```

4.4 팝업 처리

팝업 종류별 (모달팝업, 모달리스, 모달싱크, 모달윈도우)로 공통함수를 사용하여 팝업을 호출 한다.

샘플 화면 : smaple:: samplePopup.xfdl

4.4.1 팝업 종류별 차이점

팝업종류	브라우저	처리방식	callbackFunc	프레임밖 이동
modal	html5, runtime	aSync	유	불가능
modeless	html5, runtime	aSync	무	가능
modalSync	runtime	Sync	무	불가능
modalWindow	runtime	Sync	무	가능

4.4.2 팝업 옵션

옵션	설명
top	상단좌표
left	좌측좌표
width	넓이
height	높이
popupType	팝업 종류 Modal : showModal() Modeless : application.open() Modalsync : showModalSync() Modalwindow : showModalWindo
title	팝업타이틀
layered	투명윈도우

옵션	설명
opacity	투명도
autosize	자동 사이즈 조절

4.4.3 팝업 호출 방법

```
/**
 * @class 팝업오픈
 * @param {String} sPopupId - 팝업ID
 * @param {String} sUrl      - 팝업URL
 * @param {String} [oArg]   - 전달값
 * @param {String} [sPopupCallback] - 팝업콜백
 * @param {Object} [oOption] - 팝업옵션 <br>
 *     oOption.top          : 상단 좌표 <br>
 *     oOption.left         : 좌측 좌표 <br>
 *     oOption.width        : 넓이 <br>
 *     oOption.height       : 높이 <br>
 *     oOption.popupType    : 팝업종류(modal:showModal, <br>
 *                             modeless:application.open, <br>
 *                             modalsync:showModalSync, <br>
 *                             modalwindow:showModalWindow) <br>
 *     oOption.layered       : 투명 윈도우 <br>
 *     oOption.opacity      : 투명도 <br>
 *     oOption.autosize      : autosize <br>
 * @return N/A
 * @example
 * this.gfnOpenPopup(this);
 */

//modal
```

```

var sTitle = "모달팝업(Default)";

var oArg = {pvString:"abcdedd", pvNumber:555555, pvDataset:this.dsData};
var oOption = {
    top:100
    ,left:100
    ,width:700
    ,height:300
    ,popuptype:"modal"    //modal,modaless
    ,autosize:false
    ,title:sTitle
    ,resize:false
    ,titlebar:true};

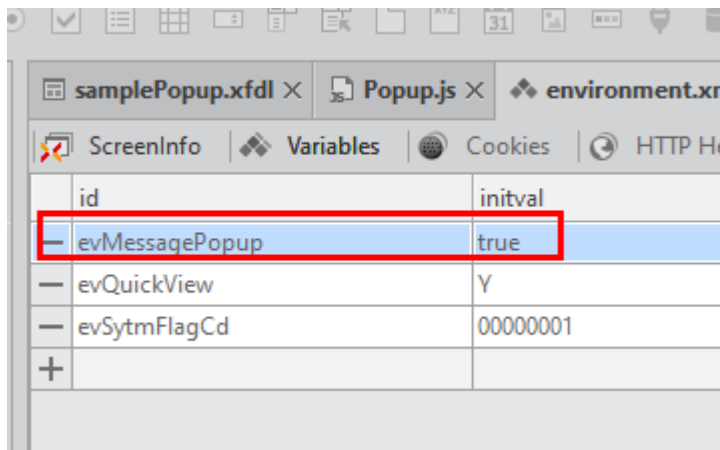
var sPopupCallBack = "fnPopupCallback";

this.gfnOpenPopup("popupModalDefault", "sample::samplePopupP.xfdl", oArg,
sPopupCallBack, oOption);

```

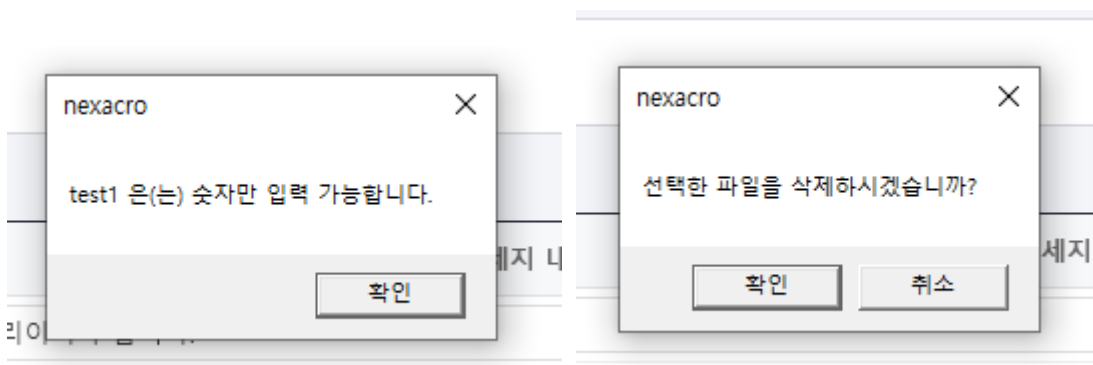
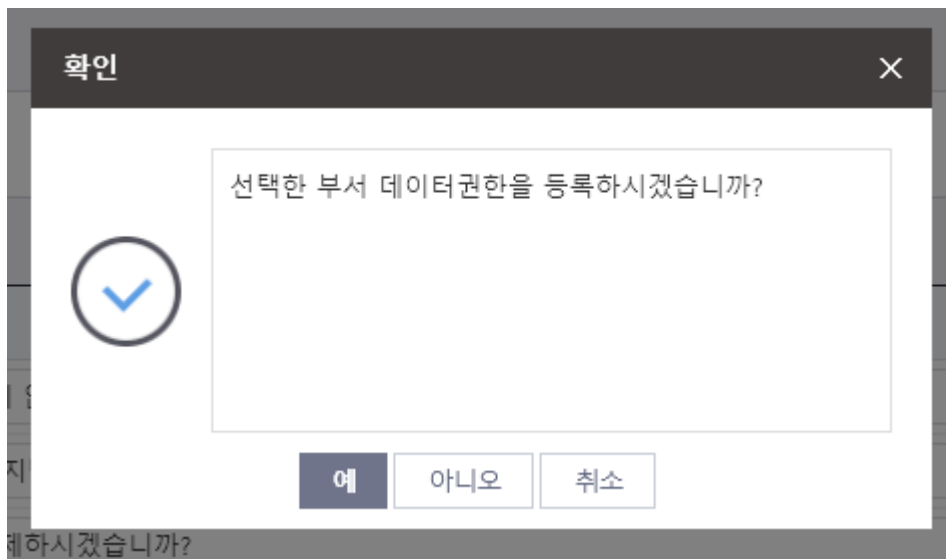
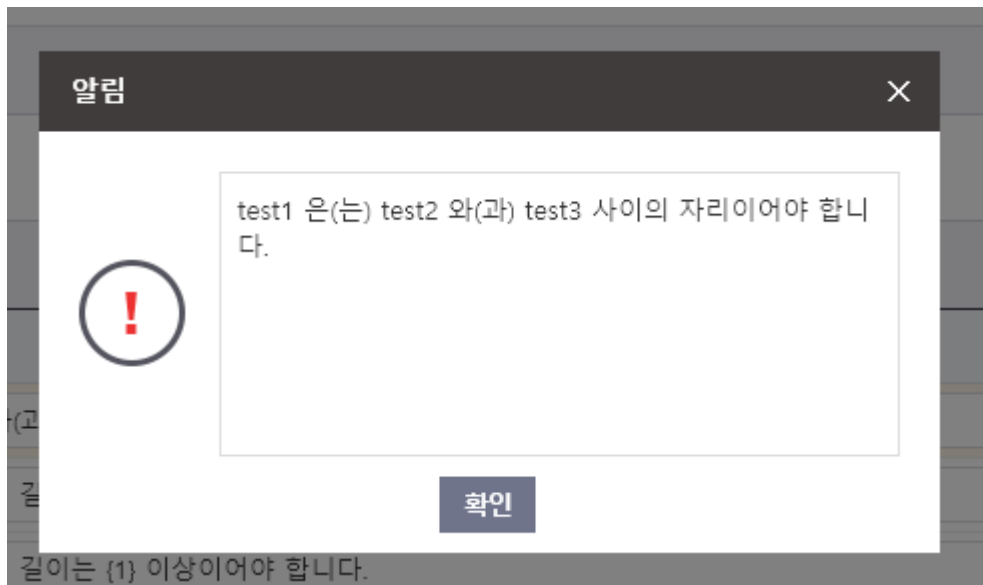
4.5 메시지

메시지 팝업을 통해 사용자에게 알림, 확인 창을 호출 한다.



Message 의 경우 system message 와 popup message 가 있고 해당 내용은 Environment Variables 에 [evMessagePopup] true/ false로 설정 할 수 있다.true 설정 시 popup message 로 메시지 창이 표현되고, false 설정 시, system 메시지로 설정된다.

샘플 화면 : sample::sampleMessage.xfdl



```
/**
 * @class 메시지팝업오픈
 * @param {String} sMsgId - 메시지ID
```

```

* @param {Array} arrArg - 메시지에 치환될 부분은 "{0~N}"이 되고 치환값은 배열로 넘김
* @param {String} [sPopId] - 팝업ID(하나의 callback함수에서 중복된 메시지 처리를 할
경우 PopId구분을 위해 unique한 ID 반드시 사용)

* @param {String} [sCallback] - 팝업콜백 (confirm성 메시지를 사용시 반드시 필요)

* @param {Array} [arrButton] - 확인 창에서 버튼의 명칭 배열

* @param {Array} [arrRtn] - 확인 창에서 버튼 클릭시 리턴할 값 배열

* @return N/A
* @example
* // {0} 항목의 최대 입력글자수를 초과하였습니다. 최대길이 : {1}

* this.gfnAlert("NAA0023", ["이름", "3자"]);

* // 저장하시겠습니까?

* this.gfnAlert("NAA0012", null, "NAA0012", "fnMsgCallback", ["예", "아니오",
"취소"], ["Y", "N", "C"]);

*/

```

4.5 정합성 (Validation) 체크

Validation check 시 4 개의 공통함수를 이용하여 룰을 설정/초기화 하고 정합성을 체크한다.

함수명	설명
gfnClearValidation	Dataset에 설정된 정합성 체크 Ruleset을 Clear한다.
gfnSetValidation	Dataset의 Column 별로 정합성 체크 Rule을 등록한다.
gfnRemoveValidataion	Dataset의 Column별로 설정된 정합성 체크 Rule을 제거 한다.

함수명	설명
gfnValidation	Dataset에 등록된 데이터 정합성 체크 Ruleset에 의해 정합성을 체크하고 이상 여부를 리턴 한다.

샘플 화면 : *sample::sampleValidation.xfdl*

```
// 정합성 체크
this.gfnClearValidation(this.dsList);

this.gfnSetValidation(this.dsList, "ID" , "아이디"
    , "required,minlength:3,maxlength:8");
=this.gfnSetValidation(this.dsList, "SSN" , "주민번호" ,
    "required,issn");
this.gfnSetValidation(this.dsList, "DATE_FROM" , "시작일자,종료일자" ,
    "required,date");
this.gfnSetValidation(this.dsList, "DATE_TO" , "종료일자,시작일자" ,
    "required,date,fromto:DATE_FROM");
this.gfnSetValidation(this.dsList, "MAX_NUM" , "금액"
    , "required,digits,maxlengthdec:7:2");
this.gfnSetValidation(this.dsList, "DECIMAL" , "최대값"
    , "required,digits,max:10");
this.gfnSetValidation(this.dsList, "COMPARE_1" , "COMPARE_1,COMPARE_2" ,
    "comparebig:COMPARE_2");

// Dataset의 변경된 Row Validation check
if (this.gfnValidation(this.dsList, "U") == false) return;

trace("Validation이 통과 되었습니다.");
```

4.6 공통버튼 사용

프로그램 관리 및 권한관리에서 공통버튼의 사용 유무를 등록 후 사용 할 수 있다. 생성된 버튼은 각 업무 화면의 cfn+버튼명 함수를 호출 한다.

The screenshot shows the '프로그램 관리' (Program Management) interface. It includes a table of programs and a detailed view of program buttons. The table has columns: 프로그램 ID, 모듈, 프로그램 명, 프로그램 유형, 프로그램 경로, and 사용 여부. The detailed view shows a grid of checkboxes for various buttons like '조회' (Search), '추가' (Add), '삭제' (Delete), etc., for each program.

```

25  };
26  //*****
27  * 공통함수영역 (cfnSearch : 조회 / cfnSave : 저장 / cfnAdd : 신규 / cfnDel : 삭제 / cfnPrint : 인쇄..)
28  *****/
29  //조회
30  this.cfnSearch = function ()
31  {
32      //TODO..
33  };
34  //저장
35  this.cfnSave = function ()
36  {
37      //TODO..
38  };
39  //추가
40  this.cfnAdd = function ()
41  {
42      //TODO..
43  };
44  //삭제
45  this.cfnDel = function ()
46  {
47      //TODO..
48  };
49  };
50  };
51  };
52  };

```

버튼명	함수명	설명
조회	cfnSearch	각 업무화면 조회조건 영역에 배치

버튼명	함수명	설명
추가	cfnAdd	
삭제	cfnDel	
저장	cfnSave	
출력	cfnPrint	사용x
엑셀	cfnExcel	사용x
초기화	cfnInit	사용x

4.7 공통버튼 추가버튼

프로그램 관리 및 권한관리에서 공통버튼의 사용 유무를 등록 후 사용 할 수 있다. 생성된 버튼은 각 업무 화면의 cfn+버튼 ID 함수를 호출 한다

The screenshot displays the 'DevPACK' application interface. A modal window titled '추가버튼 관리' (Add Button Management) is open, showing a table for adding buttons. The table has the following columns: 버튼ID (Button ID), 버튼명 (Button Name), 버튼 영문명 (Button English Name), 버튼 스타일 (Button Style), 정렬순서 (Sort Order), and 사용여부 (Usage Status). Two buttons are listed: btnAdd1 and btnAdd2. The '사용여부' column has checkboxes for '사용' (checked) and '미사용' (unchecked). The background shows the '프로그램 관리' (Program Management) screen with a list of programs and their permissions.

버튼ID	버튼명	버튼 영문명	버튼 스타일	정렬순서	사용여부
btnAdd1	추가버튼1	btnAdd1	btn_WF_Custom	1	<input checked="" type="checkbox"/>
btnAdd2	추가버튼2	btnAdd2	btn_WF_Custom	2	<input checked="" type="checkbox"/>


```
//추가버튼1
this.cfnbtnAdd1 = function ()
={
    this.gfnAlert("msg.general", ["추가버튼1입니다."]);
    return;
};

//추가버튼2
this.cfnbtnAdd2 = function ()
={
    this.gfnAlert("msg.general", ["추가버튼2입니다."]);
    return;
};
```