

# Developer Guide

**Integrated Systems**

**Shanghai Headquarters**

**iSET-DA Shanghai**

**2019-2022**

# Contents

## 0. 개요

## 1. Client 설치

## 2. Framework 구성요소

## 3. User Interface 개발

## 4. Server-Client 통신

## 5. UI 간 통신

## 6. 복수의 매개변수 전달하기

## 7. TAP BIZ Controls

## 8. Namespace Naming Rule

## Appendix 1. UI Templates

## Appendix 2. 표준 코드 작성법

## Appendix 3. 설치 문제점 및 해결방법

## 파일 릴리즈

Date	Description	Writer
2019.06.18.	Initial Draft	KWON HYUK
2019.10.11.	Adds Section 7	KWON HYUK
2022.02.08.	Modify to Integrated Systems	KWON HYUK
2022.04.06.	Adds Section 8	KWON HYUK

## 개요

이 문서는 FAPFX와 Integrated Systems 기반의 애플리케이션을 개발하는 개발자들에게 “개발표준”을 제시하기 위해 작성되었습니다. 이 문서에는 Application Setup, 클라이언트 개발, 코딩 규칙 및 Namespace 명명규칙 등을 포함합니다.

이 문서의 Section 1과 Section 2는 애플리케이션의 기본구성에 대해 설명하고 있으며, 이 부분은 초보 개발자들에게는 다소 어려울 수 있습니다. 따라서 이 문서를 읽고 있는 개발자가 팀리더라면 Section 1부터 읽기를 권장하지만, 그렇지 않다면 Section 3부터 문서를 읽어도 문제가 없습니다.

이 문서를 읽고 계신 개발리더분들은 팀원들이 해당 문서를 읽고, 해당문서의 표준에 따라 개발을 진행할 수 있도록 지원과 지도 부탁드립니다.

# 1. Client 설치

## 1.1. Framework 설치 및 Directory 구성

제공된 tapfxat\_\*\_dev\_kit.zip 파일을 프로젝트의 Root Directory에 압축 해제한다([Fig1.1-1]).

이름	수정한 날짜	유형
DB	2018-08-06 오전...	파일 폴더
FX	2018-08-06 오전...	파일 폴더
ISFAClient	2018-08-06 오전...	파일 폴더
SourceCodes	2018-08-06 오전...	파일 폴더

[Fig1.1-1]

FX Directory의 \*.exe 파일, \*.exe.config 파일, \*.config 파일을 확인한다.

ISFA.config	2018-08-06 오전 10:09	XML Configuratio...	4KB
ISFA.config.bak	2018-07-05 오후 1:09	BAK 파일	4KB
ISFA.exe	2018-07-05 오후 1:19	응용 프로그램	1,906KB

[Fig1.1-3]

[Fig1.1-3]에서 \*.exe 파일은 실행파일, \*.config 파일은 Application의 Configuration 파일, \*.exe.config 파일은 Framework의 Configuration 파일이다.

### 1.1.1. Framework Configuration 구성

FX Directory에서 \*.exe.config파일을 연다.

### 1.2.1. Environment Section

Environment Section을 구성한다. 이 Section은 Application의 동작 방식을 정의한다. Environment Section의 각 Key의 의미는 [Table1.2.1-1]과 같다.

Key	설명	비고
<b>Region</b>	Application에서 다룰 가장 큰 Key 값이다. 일반적으로 Application이 설치된 지역의 이름을 할당한다.	
<b>InstallType</b>	Application이 3Tier 환경에서 동작할 경우, " <b>CLIENT</b> "를 할당한다. 그렇지 않은 경우 " <b>SERVER</b> "를 할당한다.	

[Table1.2.1-1]

Region의 Key 값을 변경할 경우, Application Configuration 파일인 \*.Config의 Region Element 이름도 변경해 주어야 한다.

### 1.2.2. Data Source Section

Data Source Section을 구성한다. 이 Section은 Application의 Data 소스를 설정한다. 만약 Application이 Data를 연동하지 않을 경우, 해당 Section을 구성하지 않아도 된다. Data Source Section의 각 Key의 의미는 [Table1.2.2-1]와 같다.

Key	설명	비고
<b>DataSource</b>	Data 소스로 Database를 연동하는 경우, " <b>DB</b> "를 할당한다. XML 파일을 연동하는 경우, " <b>XML</b> "을 할당한다.	
<b>InfoBase</b>	Data 소스와의 연결정보를 Framework Configuration에 설정하는 경우 " <b>FX</b> "를 입력한다. 그렇지 않은 경우 " <b>APP</b> "를 입력한다.	

[Table1.1.2-1]

### 1.2.3. Database Section

Database Section을 구성한다. 이 Section은 Database와의 연결을 설정한다. 만약 Application이 Database와 통신을 하지 않거나 3Tier 환경에서 동작할 경우, 해당 Section을 구성하지 않아도 된다. Database Section의 각 Key의 의미는 [Table1.2.3-1]와 같다.

Key	설명	비고
<b>Default</b>	기본적으로 연동되는 Database 정보의 Key를 입력한다. Application이 단일 Database를 연동할 경우, " <b>DEFAULT</b> "를 할당한다.	
<b>Timeout</b>	Database로부터의 최대 응답 대기 시간을 할당한다(단위: 초).	

[Table1.2.3-1]

Database Section에서는 Application이 연동하는 Database의 수만큼 Connection Element를 추가해야 한다. Connection Element의 각 Key의 의미는 [Table1.2.3-2]과 같다.

Key	설명	비고
<b>Key</b>	해당 Connection정보의 Key 값이다. Application이 단일 Database를 연동할 경우, " <b>DEFAULT</b> "를 할당한다.	
<b>DBMS</b>	연동하는 DBMS의 종류를 할당한다. 연동하는 DBMS가 Oracle일 경우 " <b>ORACLE</b> ", SQL-SERVER일 경우 " <b>MSSQL</b> ", DB2일 경우 " <b>DB2</b> "를 할당한다. 이 외의 DBMS는 현재 버전에서 지원하지 않는다.	
<b>ConnectionString</b>	연결할 Database의 Connection 정보를 할당한다.	
<b>SelectCommandMethod</b>	Database에 Query명령을 전송할 때, 사용할 형식을 할당한다. 텍스트 형식으로 명령을 전송할 경우 " <b>TEXT</b> "를, Stored Procedure 형식으로 명령을 전송할 경우 " <b>PACKAGE</b> "를 할당한다.	
<b>ModifyCommandMethod</b>	Database에 Non-Query 명령을 전송할 때, 사용할 형식을 할당한다. 텍스트 형식으로 명령을 전송할 경우 " <b>TEXT</b> "를, Stored Procedure 형식으로 명령을 전송할 경우 " <b>PACKAGE</b> "를 할당한다.	

[Table1.2.3-2]

#### 1.2.4. XML Section

XML Section을 구성한다. 이 Section은 XML파일과의 연결을 설정한다. 만약 Application이 XML 파일을 연동하지 않거나 3Tier 환경에서 동작할 경우, 해당 Section을 구성하지 않아도 된다. XML Section의 각 Key의 의미는 [Table1.2.4-1]와 같다.

Key	설명	비고
FileName	XML 파일의 Path를 할당한다.	
RootNodeName	XML 파일의 Root Node 이름을 할당한다.	
BackupPath	XML 파일의 Backup Path를 할당한다.	

[Table1.2.4-1]

### 1.2.5. Log Section

Log Section을 구성한다. 이 Section은 Application의 Logging을 설정한다. Log Section은 각 Log 타입에 따른 Log Element로 구성되어 있다. Log Element의 각 Key의 의미는 [Table1.2.5-1]과 같다.

Key	설명	비고
Type	Log의 타입. 할당된 알파벳이 소스코드 내에서 Log 관련 코드를 작성할 때 사용된다.	
Logging	해당 타입의 Log를 사용할지 여부를 설정한다. 해당 타입의 Log를 사용할 경우 "true"를, 그렇지 않을 경우 "false"를 할당한다.	
Path	해당 타입의 Log 파일이 저장될 Directory 이름이다.	
Extension	해당 타입의 Log 파일이 사용하는 확장자이다.	
MaxSize	각 Log 파일의 크기를 설정한다(단위: Byte).	

[Table1.2.5-1]

### 1.2.6. Remote Listener Section

Remote Listener Section을 구성한다. 이 Section은 원격지 컴퓨터에서 동작하고 있는 Remote Service와의 통신 연결을 설정한다. Application의 3Tier 환경에서 동작하지 않는 경우, 이 Section을 설정할 필요가 없다. Remote Listener Section의 각 Key의 의미는 [Table1.2.6-1]과 같다.



Key	설명	비고
Multi Console	원격지 컴퓨터에 복수의 Remote Service가 동작할 경우, "true"를 할당한다. 그렇지 않을 경우 "false"를 할당한다.	

[Table1.2.6-1]

Remote Host 정보는 ListenerHost Element에서 설정한다. 이 Element의 각 Key의 의미는 [Table1.2.6-2]과 같다.

Key	설명	비고
IP	Remote Host가 동작하고 있는 컴퓨터의 IP 어드레스	
Port	Remote Host와의 통신에 사용되는 Port 번호	
MinRemoteCount	동작하는 Remote Service의 최소 개수	
RetryCount	Remote Host와의 통신에 실패했을 경우, 최대 재시도 횟수	
RetryCountInterval	Remote Host와의 통신에 실패했을 경우, 다음 재시도까지의 대기시간(단위: MS)	

[Table1.2.6-2]

각 Remote Service 정보는 Listners Element내의 Listner Element에서 설정한다. Remote Service의 개수만큼 Listener Element를 추가해야 한다. 이 Element의 각 Key의 의미는 [Table1.2.6-3]와 같다.

Key	설명	비고
No	Remote Service의 번호(고유값)	
Port	Remote Service가 통신에 사용되는 Port 번호	

[Table1.2.6-3]

### 1.2.7. App Section

App Section을 구성한다. 이 Section은 Application Launcher의 구성을 설정한다. App Section의 각 Key의 의미는 [Table1.2.7-1]과 같다.

Key	설명	비고
<b>Facility</b>	Application에서 다룰 두번째 큰 Key값이다. 일반적으로 시설물 이름을 할당한다.	
<b>UserLanguage</b>	Application이 동작할 때 기본 적용될 언어를 할당한다. 한국어일 경우 " <b>KR</b> ", 영어일 경우 " <b>EN</b> ", 중국어일 경우 " <b>CN</b> "을 할당한다. 이 밖의 언어는 현재 버전에서 지원되지 않는다.	
<b>MDIDisplayName</b>	제품의 이름을 할당한다. 이 Key에 할당된 이름이 Launcher의 제목으로 표시된다.	
<b>AppConfigFileName</b>	Application Configuration 파일 이름	
<b>AppConfigName</b>	Application Configuration 파일의 Root Node 이름	

[Table1.2.7-1]

#### 1.2.7.1. App Element

Apps Element에 App 엘리먼트를 추가하여 Application 그룹을 추가한다. App Element의 각 Key의 의미는 [Table1.2.7.1-1]과 같다.

Key	설명	비고
<b>Key</b>	Application 그룹의 Key 값이다.	
<b>Sequence</b>	Application Launcher에 해당 Application 그룹이 표시되는 순서이다.	

[Table1.2.7.1-1]

App Element에 SubApp Element를 추가하여 Application 정보를 설정할 수 있다. SubApp Element의 각 Key의 의미는 [Table1.2.7.1-2]과 같다.

Key	설명	비고
Key	Application의 Key 값이다.	
AppDirectory	해당 Application의 DLL 파일의 Path이다.	
Image	Application Launcher에 표시될 이미지의 파일 이름을 할당한다.	
HoverImage	Application의 Launcher에 표시될 Hover 이미지의 파일 이름을 할당한다.	
Argument	Application 시작시에 필요한 매개변수 값을 할당한다.	
AppConfg	Application 고유의 Configuration 파일이름을 할당한다.	
Enabled	Application을 Launcher에 표시할 것인지 여부를 할당한다. Application을 Launcher에 표시할 경우 "true"를 그렇지 않은 경우 "false"를 할당한다.	
DisplayName	Launcher와 제목표시줄에 표시될 Application의 이름을 할당한다.	

[Table1.2.7.1-2]

### 1.2.7.2. Startup UI Element

Startup UI Element를 사용하여 Application 시작시 실행될 UI를 설정할 수 있다. Startup UI Element의 각 Key의 의미는 [Table1.2.7.2-1]과 같다.

Key	설명	비고
Enabled	Application 시작시 특정 UI를 실행시킬지 여부를 지정한다. 특정 UI를 시작할 경우 "true"를 그렇지 않을 경우 "false"를 할당한다.	
StartUpMainMenu	Application이 시작될 때 실행할 UI의 메인메뉴 이름을 할당한다.	
StartUpMenu	Application이 시작될 때 실행할 UI의 메뉴 이름을 할당한다.	

[Table1.2.7.2-1]

### 1.2.8. Cross Language Section

Cross Language Section을 구성한다. 이 Section은 Application 다국어를 설정한다. Cross Language Section의 각 Key의 의미는 [Table1.2.8-1]과 같다.

Key	설명	비고
LocalFile	다국어 MDB 파일의 이름을 할당한다.	
ServerDirectory	Server상에 있는 다국어 MDB 파일의 Server Path를 할당한다.	
NeedToDownload	다국어 MDB 파일을 Server에서 다운로드 받아야 하는지 여부를 설정한다. Application Launcher 실행시에 새로운 다국어 MDB 파일을 다운로드 받아야 할 경우 "true"를, 그렇지 않은 경우 "false"를 할당한다.	
NeedToApply	다국어를 사용할지 여부를 설정한다. 다국어를 사용할 경우 "true"를, 그렇지 않은 경우 "false"를 할당한다.	

[Table1.2.8-1]

## 2. Framework 구성요소

Factory Configuration Manager는 구성에 대한 Utility Method를 제공합니다.

구성요소	설명	비고
<b>Base</b>	공용 Library, Application 구성, 통신 등 Application 개발의 기본적인 요소를 제공한다.	
<b>Fressage</b>	다국어 기능을 제공한다.	
<b>Mini</b>	File DB(MDB 등)와의 통신을 제공한다.	
<b>Model</b>	Data를 가공하여 Business 객체를 제공한다.	
<b>Remoting</b>	TCP/IP 기반으로 원격지 컴퓨터간 통신을 제공한다.	
<b>Service Base</b>	Windows Service Application 개발의 기본적인 요소를 제공한다.	
<b>UI</b>	UI 개발의 기본적인 요소를 제공한다.	
<b>UI Control</b>	UI를 개발할 때 필요한 기본적인 Control을 제공한다.	
<b>Web</b>	Web Application 개발에 필요한 기본적인 요소를 제공한다.	
<b>Workflow</b>	Workflow 기능을 제공한다.	
<b>App Base</b>	Base.Configuraition에서 제공하지 않는 Application 고유의 구성요소를 설정할 수 있도록 구성요소 설정 기능을 제공한다.	

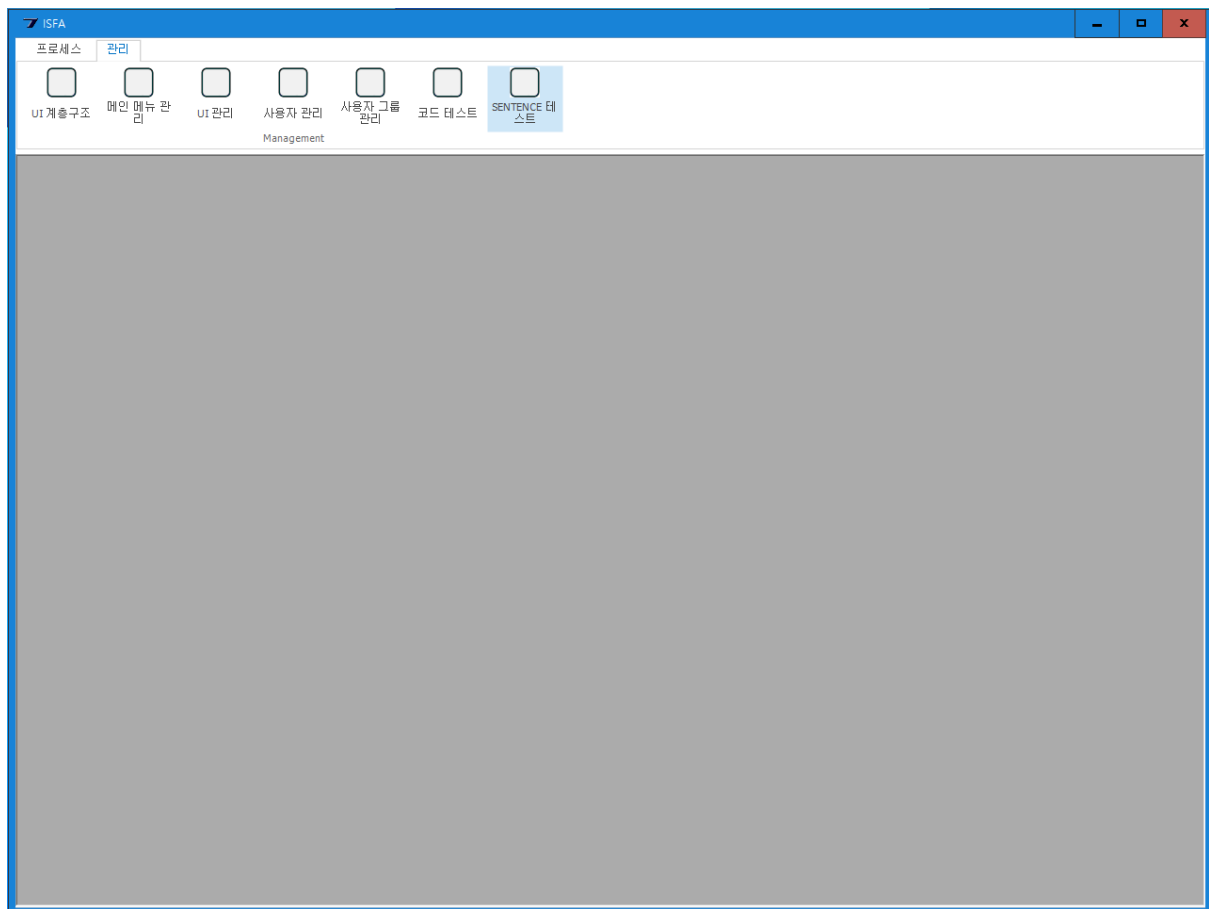
[Table2-1]

## 3. UI 개발

TAP FX의 UI는 Application MDI, Main Menu, UI의 계층구조를 갖는다. 이 Section에서는 각 계층구조의 역할을 설명하고, UI 개발에 필요한 구성요소를 설명한다.

### 3.1. Application MDI

[Fig3.1-1]은 TAP FX의 MDI이다.



[Fig3.1-1]

## 3.2. UI 만들기

TAP FX환경의 UI는 TAP.UI.UITemplate를 상속받아 제작한다. UI Template는 UI의 레이아웃이다. UI Template의 종류와 각각의 Layout은 본 문서의 Appendix 1을 참조한다. 또한 UI Template는 TAP.UI.UIBase를 상속받는다. UI Base는 UI 제작에 필요한 각종 기능들을 제공한다.

### 3.2.1. UI 제작 준비.

TAP FX 환경에서 UI를 제공하기 위해서는 기본적으로 프로젝트에 다음의 항목들을 참조해야 한다.

- TAP.App.Base
- TAP.Base
- TAP.Base.Configuration
- TAP.Fresssage
- TAP.Mini
- TAP.Models
- TAP.UI
- TAP.UIControls
- TAP.UIControls.BasicControls
- TAP.UIConstrols.Sheets

3.2.1.2. 프로젝트에 새로운 Windows Form을 추가한다.

3.2.1.3. 새로운 Windows Form이 UI Template를 사용하도록 Class의 상속 파트를 수정한다. 이 예에서는 UI가 UI Template L4T1을 사용하도록 했다([Fig3.2.1.3-1]).

```
2 references
public partial class Form2 : TAP.UI.UITemplate4.UITemplateL4T1
{
    0 references
    public Form2()
    {
        InitializeComponent();
    }
}
```

[Fig3.2.1.3-1]

3.2.1.4. 새로운 Windows Form의 초기화를 위해 Initialize 메서드를 Override 해준다. Override한 Initialize 메서드는 최우선적으로 UI Base의 InitializeUI 메서드를 호출해야 한다. UI Base의 InitializeUI 메서드 호출 구문 아래에 새로 제작할 Windows Form 고유의 초기화 코드를 작성한다([Fig3.2.1.4-1]).

```
protected override void Initialize()
{
    #region Initialize

    try
    {
        base.InitializeUI();

        //TO DO: Adds your initializing codes
    }
    catch (System.Exception ex)
    {
        throw ex;
    }

    #endregion
}
```

[Fig3.2.1.4-1]

추가한 Initialize 메서드는 UI가 실행될 때, 가장 먼저 호출되는 메서드이며, UI Base의 Initialize UI는 UI 내 각 Control의 다국어 지원을 지원하는 등 기본적인 UI 초기화 작업을 진행한다.

### 3.2.2. Callback 형식의 비동기 호출 사용하기

TAP FX 환경의 UI는 Callback 비동기 호출 기능을 제공한다.



3.2.2.1. 첫번째 호출될 메서드를 만든다. 이 메서드는 반드시 반환값이 존재해야 한다. [Fig3.2.2.1-1]은 List<Model> 형식을 반환한다.

```
virtual public List<Model> LoadData()
{
    #region Load Data

    Lot tmpLot = null;

    try
    {
        tmpLot = new Lot((BIZDefaultInfo)this._defaultInfo);
        this._modelList = tmpLot.LoadModelHistory();
        return _modelList;
    }
    catch (System.Exception ex)
    {
        throw ex;
    }

    #endregion
}
```

[Fig3.2.2.1-1]

[Fig3.2.2.1-1]의 코드는 반환값이 존재하는 메서드를 보여주기 위한 예이다. 내부의 코드들은 지금 이해하지 못해도 상관이 없다.

3.2.2.2. 두번째 호출된 메서드(Callback 메서드)를 만든다. 이 메서드는 3.2.2.1에서 만든 반환값 형식을 매개 변수로 가져야 한다.

### 3.3. 다국어 설정

#### 3.3.1. 다국어 설정 파일

TAP FX의 구성요소인 Fessage Code Collection을 통해 다국어를 설정할 수 있다. Fessage Code Collection은 MDB 파일 형식이며, Framework Directory의 "mnls" Directory에 위치한다. 파일의 이름은 Framework Configuration Cross Language Section의 LocalFile Key의 값과 일치해야 한다.

다국어 설정 파일의 내부 구조는 [Fig3.3.1-1]과 같다.

ID	CODE	POS	EN	KR	CN	CC
788	ARGUMENT	NOUN	argument	매개변수	参数	ETC
399	ARITHMETIC	NOUN	arithmetic	산술	算术	ETC
553	ARITHMETIC OPERATOR	IDOM	arithmetic operator	산술 연산자	算术运算符	ETC
608	ASSEMBLY	NOUN	assembly	어셈블리	汇编	ETC
457	ASSIGNER	NOUN	assigner	할당자	分配者	ETC

[Fig3.3.1-1]

각 컬럼의 의미는 [Table3.3.1-1]과 같다.

컬럼	설명	비고
CODE	Fressage Code. UI를 제작할 때, 이 코드를 사용하여 Control의 Text 값을 할당한다.	
POS	해당 코드의 품사. Fressage에서 사용되는 품사는 [Table3.3.1-2]를 참조한다.	
EN	해당 코드의 영어 단어(구)	
KR	해당 코드의 한국어 단어(구)	
CN	해당 코드의 중국어 단어(구)	
CC	해당 단어의 특성(코드의 품사가 NOUN인 경우에만 할당)	

[Table3.1.1-1]

품사	설명	비고
NOUN	명사	
ADJECTIVE	형용사	
VERB	동사	
ACRONYM	예약어	다국어로 컨버트되지 않음.
IDOM	숙어. 두 개 이상의 단어로 구성된 코드	
ADVERB	부사	

[Table3.1.1-2]

### 3.3.2. Code Phrase(코드구) 만들기

Fressage의 기본 Converting 단위는 한 개 이상의 단어로 구성된 Code Phrase이다. 사용자는 기초 영문법에 의거하여 Code Phrase를 구성한다. [Table3.1.2-1]은 Code Phrase가 각 언어로 Converting된 몇 가지 예이다.

Code Phrase	영어	한국어	중국어	비고
SEARCH	Search	검색	调查	
SEARCH RECIPE	Search recipe	레시피 검색	调查配方	
MOVE PITCH	Move pitch	Pitch 이동	移动 Pitch	
CURRENT CELL	Current CELL	현재 CELL	现在 CELL	
SCAN FIELD	Scan field	필드 스캔	扫描 Field	

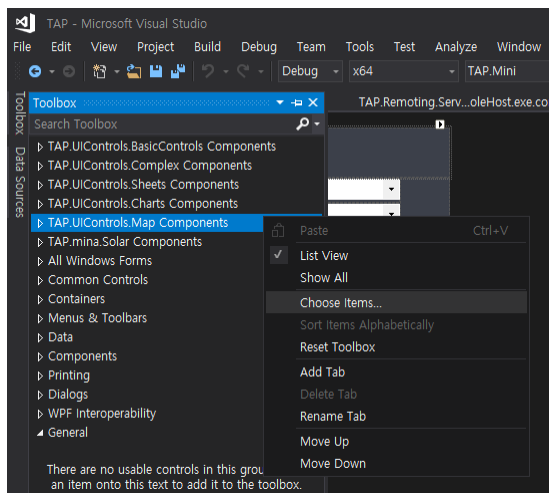
[Table3.1.2-1]

## 3.4. TAP UI Control 사용하기

TAP FX는 Visual Studio가 기본적으로 제공하는 UI Control 또는 Third Party Control을 조금 더 사용하기 편리하도록 Wrapping하였다.

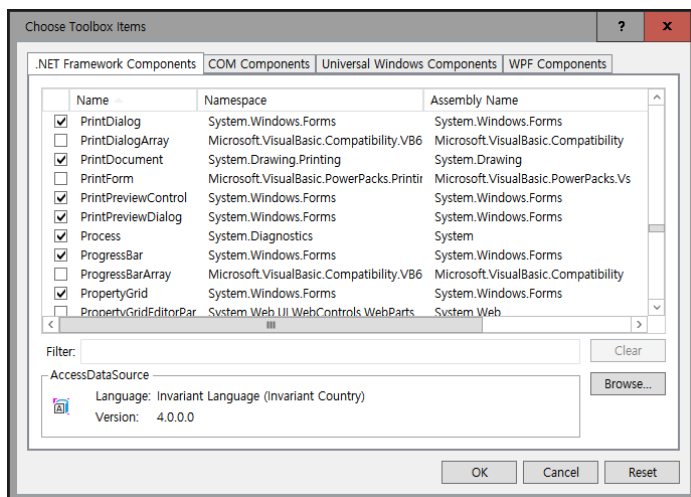
### 3.4.1. Visual Studio에 TAP UI Control 추가하기

3.4.1.1. [Fig3.4.1.1-1]처럼 Visual Studio의 Toolbox에서 마우스 오른쪽 버튼을 클릭한 후, Context Menu에서 [Choose Items...]를 선택한다.



[Fig 3.4.1.1-1]

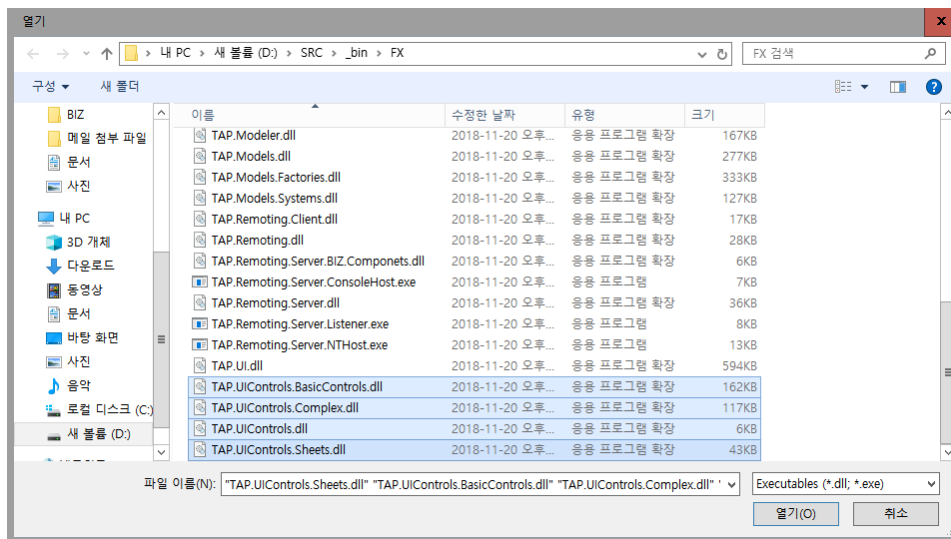
3.4.1.2. Choose Toolbox Items 창의 초기화가 완료되면, [Browse]버튼을 클릭한다.



[Fig 3.4.1.2-1]

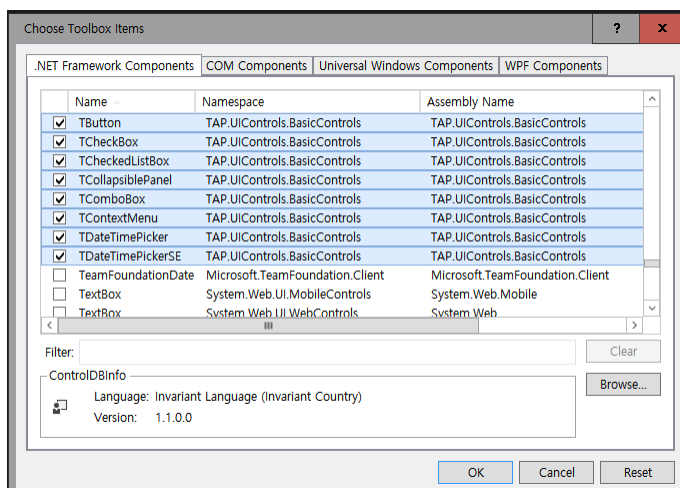
3.4.1.3. TAP FX Client가 설치되어 있는 폴더에서 다음의 DLL을 선택한 후, [확인]을 클릭한다.

- TAP.UIControls.dll
- TAP.UIControls.BasicControls.dll
- TAP.UIControls.Sheet.dll
- TAP.UIControls.Complex.dll



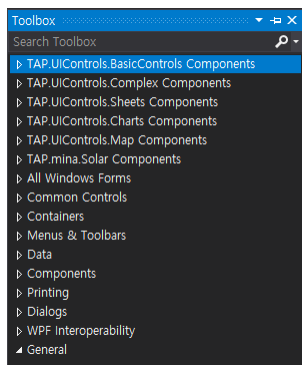
[Fig3.4.1.3-1]

3.4.1.4. Choose Toolbox Items 창에서 TAP.UIConrols\*의 DLL 들을 선택한 후, [확인]버튼을 클릭한다.



[Fig3.4.1.4-1]

3.4.1.5. Toolbox에 TAP UI Control들이 로드 되었는지 확인한다.

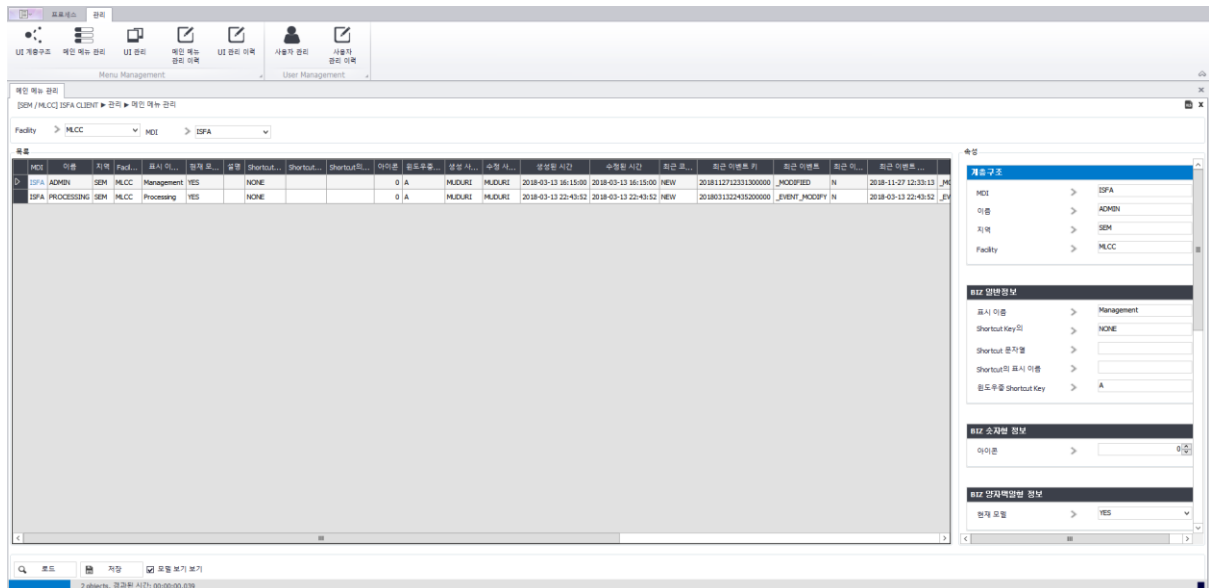


[Fig3.4.1.5-1]

## 3.5. 메뉴관리

### 3.5.1. 메인메뉴 관리

메인메뉴 관리는 [관리]-[메인메뉴관리]에서 진행한다(Fig3.5.1-1).



[Fig3.5.1-1]

#### 3.5.1.1. 메인메뉴 추가

메인메뉴의 추가/삭제/수정 작업은 메인메뉴 관리 UI 우측의 [속성]창에서 진행한다.

#### 3.5.1.1.1. 계층구조

[계층구조]에서 해당 메인메뉴가 속한 MDI 이름, 메인메뉴의 이름, Region, Facility를 입력한다([Fig 3.5.1.1.1-1]).

계층구조		
MDI	>	ISFA
이름	>	ADMIN
지역	>	SEM
Facility	>	MLCC

[Fig3.5.1.1.1-1]

#### 3.5.1.1.2. 일반정보

[일반정보]에는 메인메뉴의 표시이름(Display Name)을 입력한다([Fig3.5.1.1.2-1]).

BIZ 일반정보		
표시 이름	>	Management
Shortcut Key의	>	NONE
Shortcut 문자열	>	
Shortcut의 표시 이름	>	
윈도우중 Shortcut Key	>	A

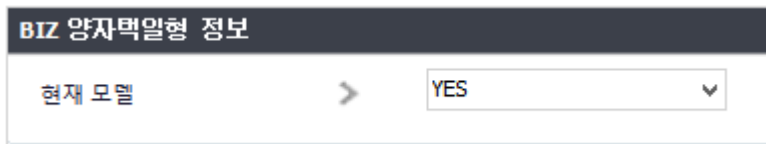
[Fig3.5.1.1.2-1]

#### 3.5.1.1.3. 저장

모든 정보 입력이 완료되었으면, UI 좌측 하단의 [저장]버튼을 클릭하여 저장한다.

#### 3.5.1.2. 메인 메뉴 제거

메인메뉴 제거시에는 [속성]의 [BIZ 양자택일형 정보]에서 [현재모델]의 값을 "NO"로 변경한 후, 저장 버튼을 클릭한다([Fig3.5.1.2-1]).

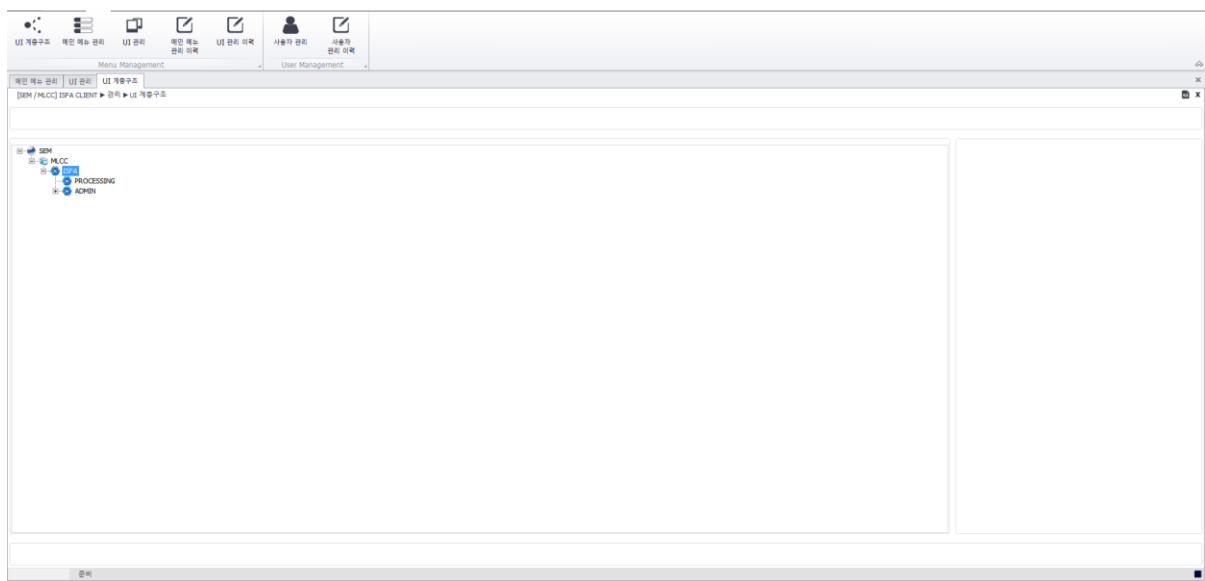


The dialog box titled "BIZ 양자택일형 정보" (BIZ Exclusive Information) contains a label "현재 모델" (Current Model) followed by a right-pointing arrow and a dropdown menu currently showing "YES".

[Fig3.5.1.2-1]

### 3.5.1.3. 메인메뉴 순서 설정

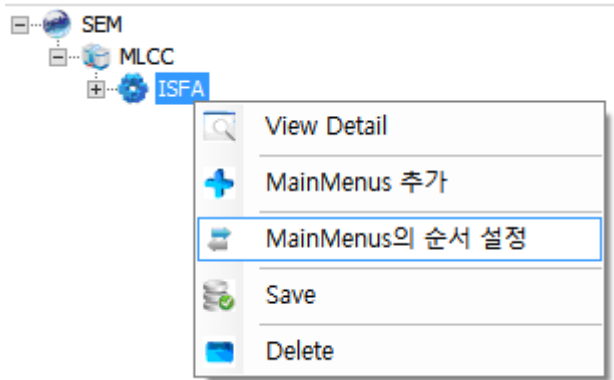
메인메뉴 순서는 [관리]-[UI계층구조]에서 진행한다([Fig3.5.1.3-1]).



[Fig3.5.1.3-1]

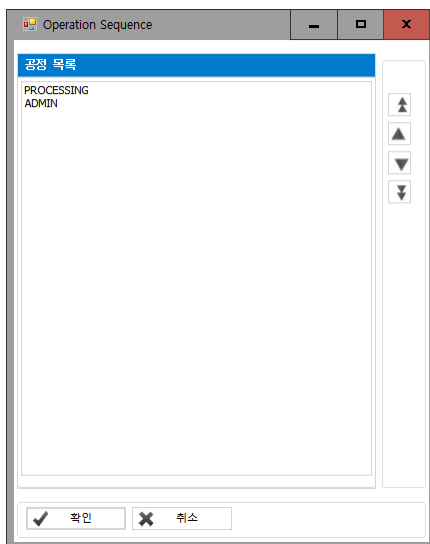
트리뷰에서 MDI이름을 선택한 후, 마우스 오른쪽 클릭하여 나타나는 컨텍스트 메뉴에서 [MainMenus의 순서 설정]을 선택한다([Fig3.5.1.3-2]).





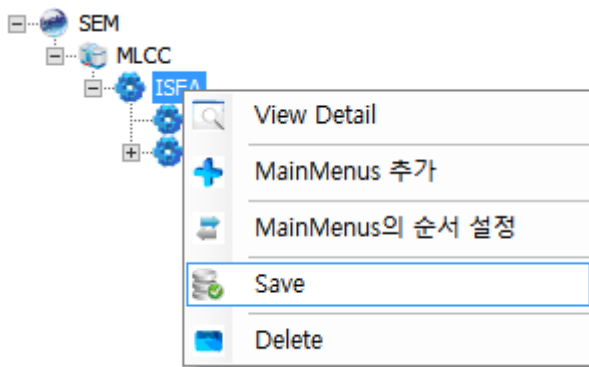
[Fig3.5.1.3-2]

[Fig3.5.1.3-3]과 같은 순서 설정 창에서 우측의 화살표를 사용하여 순서를 설정한다.



[Fig3.5.1.3-3]

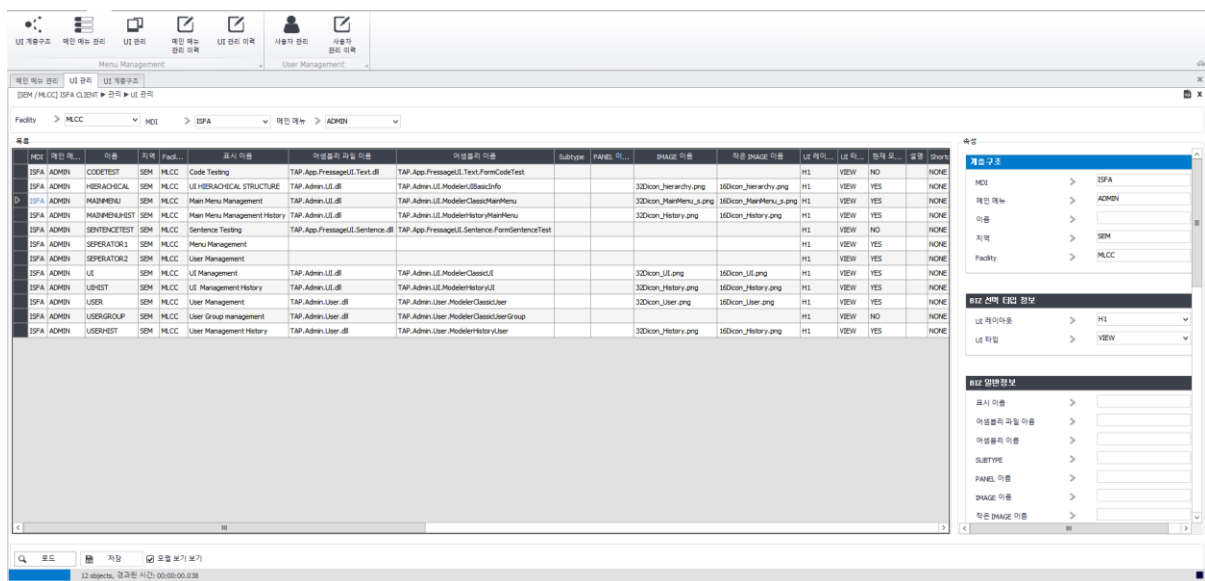
순서 설정이 완료되었으며, MDI이름을 선택한 후, 마우스 클릭하여 나타나는 컨텍스트 메뉴에서 [Save]를 선택한다([Fig3.5.1.3.-4])



[Fig3.5.1.3-4]

### 3.5.2. 메뉴 관리

메인메뉴 관리는 [관리]-[UI관리]에서 진행한다(Fig3.5.2-1).



[Fig3.5.2-1]

#### 3.5.2.1. 메뉴 추가

메뉴의 추가/삭제/수정 작업은 UI 관리 UI 우측의 [속성]창에서 진행한다.

##### 3.5.2.1.1. 계층구조

[계층구조]에서 해당 메인메뉴가 속한 MDI 이름, 메인메뉴의 이름, Region, Facility를 입력한다([Fig 3.5.1.1.1-1]).

계층구조		
MDI	>	ISFA
메인 메뉴	>	ADMIN
이름	>	UI
지역	>	SEM
Facility	>	MLCC

[Fig3.5.2.1.1-1]

### 3.5.2.1.2. 일반정보

일반정보창에 입력해야 할 정보는 [Table3.5.2.1.2-1]과 같다.

이름	설명	비고
표시이름	UI의 표시이름(Display Name)	
어셈블리 파일이름	UI를 포함하고 있는 어셈블리 파일이름	확장자를 포함해서 입력해야 함
어셈블리이름	UI의 어셈블리 이름	
IMAGE 이름	메뉴창에 나타날 이미지의 이름	해당 이미지는 Application 폴더내 [IMAGES]/[MENU]/[BIG] 폴더내에 존재해야 한다.
작은 IMAGE 이름	메뉴창에 나타날 작은 이미지의 이름	해당 이미지는 Application 폴더내 [IMAGES]/[MENU]/[SMALL] 폴더내에 존재해야 한다

[Table3.5.2.1.2-1]

[Fig3.5.2.1.2-1]은 일반정보를 구성한 화면이다.

BIZ 일반정보		
표시 이름	>	UI Management
어셈블리 파일 이름	>	TAP.Admin.UI.dll
어셈블리 이름	>	TAP.Admin.UI.ModelerClassidU
SUBTYPE	>	
PANEL 이름	>	
IMAGE 이름	>	32Dicon_UI.png
작은 IMAGE 이름	>	16Dicon_UI.png
Shortcut Key의	>	NONE
Shortcut 문자열	>	
Shortcut의 표시 이름	>	
윈도우중 Shortcut Key	>	A

[Fig3.5.2.1.2-1]

### 3.5.2.1.3. 저장

모든 정보 입력이 완료되었으면, UI 좌측 하단의 [저장]버튼을 클릭하여 저장한다.

### 3.5.2.2. 메뉴 제거

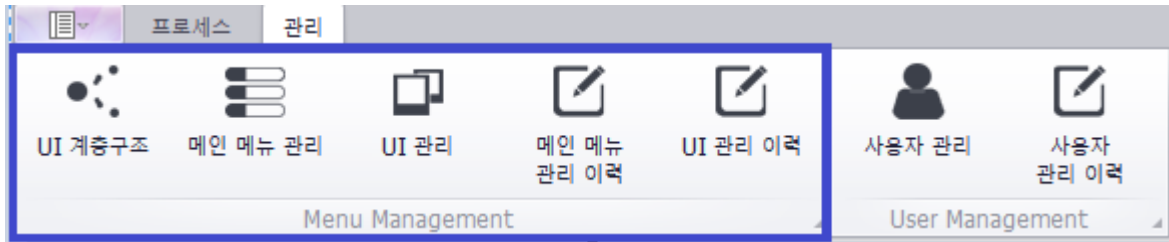
메뉴 제거시에는 [속성]의 [BIZ 양자택일형 정보]에서 [현재모델]의 값을 "NO"로 변경한 후, 저장 버튼을 클릭한다([Fig3.5.2.2-1]).

BIZ 양자택일형 정보		
현재 모델	>	YES ▼

[Fig3.5.2.2-1]

### 3.5.2.3. Page Group 설정

[Fig3.5.2.3-1]과 같이 Page Group을 설정하기 위해서는 "SEPERATOR"를 추가해야 한다.



[Fig3.5.2.3-1]

“SEPERATOR”를 추가하기 위해 [Fig3.5.2.3-1]과 같이 [계층구조]의 [이름]항목에 “SEPEARTOR\*”을 입력한다.

계층구조		
MDI	>	ISFA
메인 메뉴	>	ADMIN
이름	>	SEPERATOR1
지역	>	SEM
Facility	>	MLCC

[Fig3.5.2.3-2]

그리고 나서, [일반정보]항목의 [표시이름]에 해당 Page Group이름을 입력한다([Fig3.5.2.3-3]).

BIZ 일반정보		
표시 이름	>	Menu Management
어셈블리 파일 이름	>	
어셈블리 이름	>	
SUBTYPE	>	
PANEL 이름	>	
IMAGE 이름	>	
작은 IMAGE 이름	>	
Shortcut Key의	>	NONE
Shortcut 문자열	>	
Shortcut의 표시 이름	>	
윈도우중 Shortcut Key	>	A

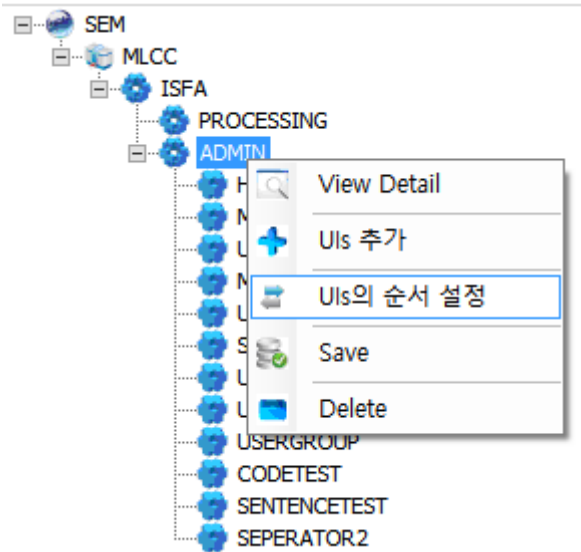
[Fig3.5.2.3-3]

모든 정보를 입력하였으면, UI 좌측 하단의 [저장]버튼을 클릭한다. 모든 "SEPERATOR"는 해당 Page Group의 맨 뒤에 위치해야 한다.

#### 3.5.2.4. 메뉴 순서 설정

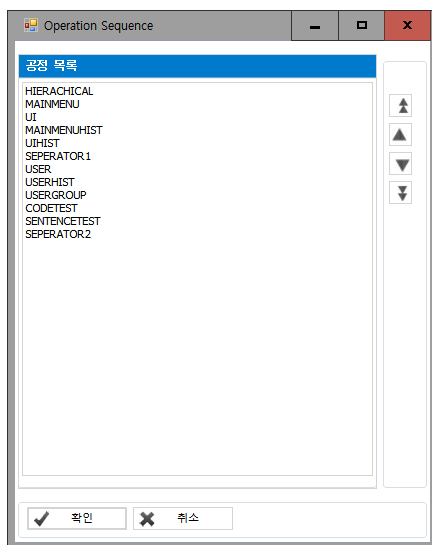
메뉴 순서는 [관리]-[UI계층구조]에서 진행한다.

트리뷰에서 메인메뉴 이름을 선택한 후, 마우스 오른쪽 클릭하여 나타나는 컨텍스트 메뉴에서 [UIs의 순서 설정]을 선택한다([Fig3.5.2.4-1]).



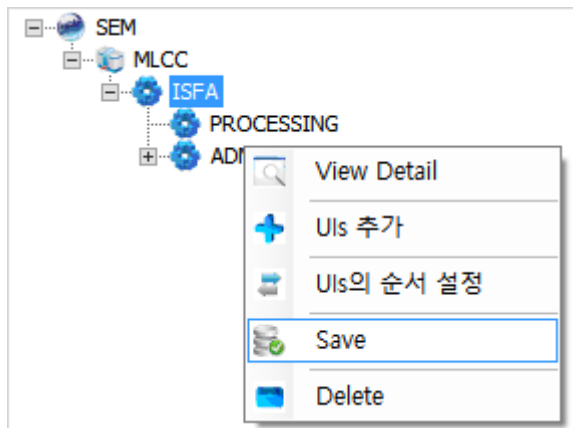
[Fig3.5.2.4-1]

[Fig3.5.1.3-2]과 같은 순서 설정 창에서 우측의 화살표를 사용하여 순서를 설정한다.



[Fig3.5.2.4-2]

순서 설정이 완료되었으며, 메인메뉴 이름을 선택한 후, 마우스 클릭하여 나타나는 컨텍스트 메뉴에서 [Save]를 선택한다([Fig3.5.2.4.-3])



[Fig3.5.2.4.-3]



## 4. Server-클라이언트 통신

TAP FX는 클라이언트가 DBMS와 직접 통신하는 2Tier 방식과, 원격지의 Host를 통해 DBMS와 통신하는 3Tier 방식을 모두 지원한다. 이 Section에서는 원격지의 Host를 설정하여, 3Tier 방식으로 DBMS와 통신하는 방법에 대해 설명한다.

### 4.1. Host 설치

Server 역할을 수행할 원격지 컴퓨터 또는 로컬 컴퓨터에 tapfxat\_svr\_dev\_kit.zip 파일의 압축을 해제한다.

“TAP.Remoting.Server.Listener.exe.config” 파일을 열어 Database Section을 수정한다.

```
<!-- Database-->
<!-- Developer must edit this section!!!-->
<DatabaseSection Default="DEFAULT" Timeout="500">
  <Connection Key="DEFAULT" DBMS="MSSQL" DTC = "true" ConnectionString="Data source=LAPTOP-UVAT3SPL\TAP_TEST;Initial Catalog=TAPSQL;Max
</DatabaseSection>
```

[Fig4.1-1]

### 4.2. 개발용 Console Host 사용하기

#### 4.2.1. Port 설정

“TAP.Remoting.Server.ConsoleHost.exe.config”파일의 Remote Listener Section을 [Fig4.2.1-1]과 같이 설정한다.

이 때, Listener Host Element와 Listener Element의 Port키에 할당된 Port 번호는 방화벽이 해제되어 있어야 한다. 만약, 보안정책상 해당 Port에 대해 방화벽을 해제할 수 없다면, 허용된 Port를 사용하도록 Key의 값을 변경해 주어야 한다.

현재 버전에서 한 개의 Console Host는 최대 다섯 개의 Listener를 관리할 수 있다. Listener Element의 EnabledKey의 값에 “true”를 할당 할 경우, 해당 Listener가 활성화 된다.

```
<!-- Listener-->
<!-- Developer must edit this section!!!-->
<RemotelListenerSection MultiConsole = "true">
  <ListenerHost IP="localhost" Port="9040" MinRemoteCount="1" RetryCount="3" RetryInterval="1500" />
  <ListenerWatcher Interval="1000" />
  <Emergency Key="F8FF170E-0046-42af-A555-09B0BF96AF6F" />
  <Listeners>
    <Listener No="1" Port="9001" Enabled="true"/>
    <Listener No="2" Port="9002" Enabled="false"/>
    <Listener No="3" Port="9003" Enabled="false"/>
    <Listener No="4" Port="9004" Enabled="false"/>
    <Listener No="5" Port="9005" Enabled="false"/>
  </Listeners>
</RemotelListenerSection>
```

[Fig4.2.1-1]

#### 4.2.2. Environment Section 수정

[Fig4.2.2-1]과 같이 "McWorksClient.exe.config" 파일의 Environment Section의 InstallTypeKey의 값을 "CLIENT"로 변경해 준다.

```
<!-- Environment Section-->
<!-- Developer must edit this section!!!-->
<EnvironmentSection Region ="WX" ExecutePath="" InstallType="CLIENT" />
```

[Fig4.2.2-1]

#### 4.2.3. Remote Adapter Section 수정

[Fig4.2.2-1]과 같이 "McWorksClient.exe.config" 파일의 Remote Adapter Section의 HostIPKey의 값을 "localhost"로 변경해 준다. Remote Adapter Section의 HostPortKey의 값과 LocalPortKey의 값에 할당되어 있는 Port는 방화벽에서 해제되어 있어야 한다.

```
<!--Remote Section-->
<!-- Developer must edit this section!!!-->
<RemoteAdapterSection HostIP="localhost" HostPort="9040" IsDebugMode="true" TestDBCode="" LocalPort ="9041"/>
```

[Fig4.2.2-1]

만약, 테스트 환경의 보안규정에 따라 해당 Port에 대해 방화벽을 해제할 수 없다면, Port를 변경해 주어야 한다. 이 때 Port를 변경할 경우, "TAP.Remoting.Server.ConsoleHost.exe.config" 파일과 "TAP.Remoting.Server.Listener.exe.config" 파일의 Remote Listener Section내 Port번호도 동일하게 변경해 주어야 한다.

#### 4.2.4. 개발용 Console Host 실행

"TAP.Remoting.Server.ConsoleHost.exe" 파일을 실행한다.

```

*****
***** TAP Console Tester Ver 1.0 base on TAP FX Nothern Star R2 *
***** Testing for Remoting *
***** Title: 'Testing host' *
***** Copyright (c) iSET-DA Shanghai Co., Ltd All rights reserved *
*****
command>

```

[Fig4.2.4-1]

[Fig4.2.4-1]과 같은 Console Program이 실행되면, 엔터를 눌러 Listener를 실행시킨다. Listener가 실행되면, 또 다른 Listener Console Program이 실행되고([Fig4.2.4-2], 개발용 ConsoleHost에는 Listener 실행정보가 표시된다([Fig4.2.4-3]).

```

2018-03-21 16:50:04.986: Listner01 - 192.168.1.105:Listener Started. RemoteNo_0001 PortNo_9001
2018-03-21 16:50:05.049: Listner01 - 192.168.1.105: URI 'TAP.Remoting.Server.Listener.bin' was published.

```

[Fig4.2.4-2]

```

*****
***** TAP Console Tester Ver 1.0 base on TAP FX Nothern Star R2 *
***** Testing for Remoting *
***** Title: 'Testing host' *
***** Copyright (c) iSET-DA Shanghai Co., Ltd All rights reserved *
*****
command>
2018-03-21 16:50:04.627: Remoting - 192.168.1.105:TAP Remoting Host Started.
2018-03-21 16:50:04.783 TAP Listener started: PID_12536 SVC_01 PORT_9001
2018-03-21 16:50:04.783: Start - 192.168.1.105:Remote Agent Started

```

[Fig4.2.4-3]

#### 4.2.5. 클라이언트 실행

클라이언트 실행파일(McWorksClient.exe)을 실행시키면, Login 창이 실행되고, Listener Console에 [Fig4.2.5-1] 같이 실행 Log가 표시된다.

```
2018-03-21 17:09:35.828: Listener01 - 192.168.1.105:Listener Started. RemoteNo_0001 PortNo_9001
2018-03-21 17:09:35.937: Listener01 - 192.168.1.105: URI 'TAP.Remoting.Server.Listener.bin' was published.
2018-03-21 17:10:10.904: .ctor - 192.168.1.105:Remote Broker (18136189) was created.
2018-03-21 17:10:10.967: Execute - 192.168.1.105:Remote Broker (18136189) was called.
2018-03-21 17:10:11.029: Create - 192.168.1.105:Remote type 'DB_COMMAND' has been called.
2018-03-21 17:10:11.061: ExecuteMessage - 192.168.1.105:Remote 'TAP.Remoting.Server.Biz.BIZDBComponent' ready to execute.
2018-03-21 17:10:11.186: SelectText - : SELECT * FROM TAPTCODES
WHERE CATEGORY = 'FACILITY'
AND SUBCATEGORY = 'WX'
AND ISALIVE = 'YES'
```

[Fig4.2.5-1]

### 4.3. Server측 DB 메서드 호출하기

Server측 DB 메서드를 호출하기 위해서는 TAPData.Client.dll을 참조추가 해야 한다.

TAPData.Client.DataClient의 Instance가 제공하는 메서드를 사용하여 Server측 DB 메서드를 호출한다.

#### 4.3.1. Query 실행하기

[코드4.3.1.1-1]은 개발자의 Query문을 실행하는 예제이다. TAPData.Client.DataClient의 Instance를 생성한 후, 해당 Instance의 SelectData 메서드를 호출한다. 결과값은 DataSet 형식으로 반환된다.

```
string tmpSql = "SELECT * FROM TAPUTUSERS";
TAP.Data.Client.DataClient tmpDBC = new Data.Client.DataClient();
DataSet ds = tmpDBC.SelectData(tmpSql, "TAPUTUSES");
```

[Code4.3.1-1]

#### 4.3.2. Non-Query 실행하기

[코드4.3.2-1]은 개발자의 Non-Query문을 실행하는 예제이다. TAPData.Client.DataClient의 Instance를 생성한 후, 해당 Instance의 ModifyData 메서드를 호출한다. 결과값은 int 형식으로 반환된다.

```
string tmpNonQuery = "UPDATE TAPUTUSERS SET MOBILENO = '123456789'";
TAP.Data.Client.DataClient tmpDBC = new Data.Client.DataClient();
tmpDBC.ModifyData(tmpNonQuery);
```

[Code4.3.2-1]

### 4.3.3. Transaction 실행하기

[코드4.3.2-1]은 개발자의 Non-Query문을 Transaction으로 실행하는 예제이다. TAPData.Client.DataClient의 Instance를 생성한 후, Transaction 처리하고자 하는 Non-Query문들을 String 타입의 List에 적재하고, ModifyData 메서드를 호출한다.

```
string tmpNonQuery1 = "UPDATE TAPUTUSERS SET CONTACTNO = '123456789'";
string tmpNonQuery2 = "UPDATE TAPUTUSERS SET MAILADDRESS = 'hkwon@iset-da.com'";

List<string> tmpNonQueries = new List<string>();
tmpNonQueries.Add(tmpNonQuery1);
tmpNonQueries.Add(tmpNonQuery2);

TAP.Data.Client.DataClient tmpDBC = new Data.Client.DataClient();
tmpDBC.ModifyData(tmpNonQueries);
```

[Code4.3.2-1]

## 4.4. Server Host 구성하기

### 4.4.1. Server Host 설치

SC.exe 명령을 사용하여 Server Host 프로그램을 설치한다.

4.4.1.1. 명령 프롬프트를 관리자 권한으로 실행한다.

4.4.1.2. 명령 프롬프트에 다음의 명령어 구문을 입력한다.

```
sc create [Service이름] binPath= [프로그램 Path]
```

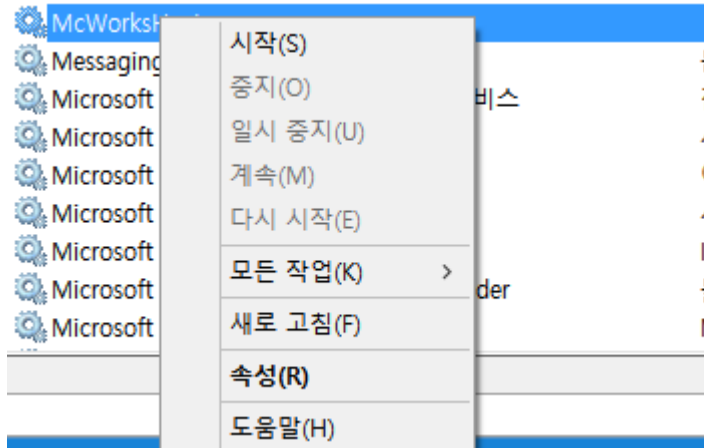
[Code4.4.1.2-1]

[Fig4.4.1.2-1]은 명령 프롬프트에서 sc.exe 명령을 사용하여, Host를 설치하는 예제이다.

```
C:\WINDOWS\system32>sc create "McWorksHost" binPath= "G:\BIZ\Source_Codes\FX\McWorksHost.exe"
```

[Fig4.4.1.2-1]

4.4.1.3. [Fig4.4.1.3-1]과 같이 Service 관리자를 실행하여 McWorksHost Service를 시작한다.



[Fig4.4.1.3-1]

#### 4.4.2. 클라이언트 구성 변경

[Fig4.4.2-1]과 같이 "McWorksClient.exe.config" 파일의 Remote Adapter Section의 HostIPKey의 값을 Host Program이 설치된 Server의 IP로 변경해 준다.

```
<!--Remote Section-->
<!-- Developer must edit this section!!!-->
<RemoteAdapterSection HostIP="172.27.1.115" HostPort="9040" IsDebugMode="true" TestDBCode="" LocalPort ="9041"/>
```

#### 4.4.3. 클라이언트 실행 및 Log 확인

클라이언트 실행파일(McWorksClient.exe)을 실행시키면, Login 창이 실행된다.

Server의 동작 상태는 Log 파일을 통해 확인할 수 있다. Log 파일은 Host Program 설치 Directory의 "logWRemotingWremoteW"Directory의 하위 Directory에 저장된다([Fig4.4.3-1])

Error	2018-03-26 오전...	파일 폴더
Info	2018-06-04 오후...	파일 폴더
Sql	2018-06-04 오후...	파일 폴더
Trace	2018-06-04 오후...	파일 폴더

[Fig4.4.3-1]



## 5. UI간 통신

TAP.UI.UIHelpBase.OpenAndCommand 메서드를 사용하여, 특정 UI에서 다른 UI를 로드 하거나, 다른 UI의 특정 메서드를 호출하는 것이 가능하다. [Fig5.1]은 OpenAndCommand 메서드의 Signature이다.

```
public void OpenAndCommand(string mainMenu, string menu, ArgumentPack arguments)
```

[Fig5.1]

OpenAndCommand 메서드의 각 매개 변수의 의미는 [Table5.1]과 같다

매개변수	설명	비고
mainMenu	호출 대상 UI의 메인 메뉴 이름	
menu	호출 대상 UI의 메뉴 이름	
arguments	호출 대상 UI를 구동 시키기 위한 매개변수.	

[Table5.1]

이 Section에서는 UI간 통신을 하기 위한 기술적 방법을 설명한다.

### 5.1. OpenAndCommand 메서드 작성하기

[Fig5.1.1]은 OpenAndCommand 메서드를 사용하여 특정 UI를 로드하고, 해당 UI에 특정 동작을 요청한다 ([Fig5.1.1]의 ArgumentPack은 본 문서의 Section 6를 참조한다).

```
ArgumentPack tmp = null;
tmp = new ArgumentPack();
tmp.AddArgument(BIZModel._COLUMN_NAME_LOT, typeof(string), tmpLotName);
TAP.UI.UICallBase.Instance.OpenAndCommand("INQUIRY", "LOTHIST", tmp);
```

[Fig5.1.1]



## 5.2. ExecuteCommand 메서드 작성하기

TAP.UI.UIHelpBase.OpenAndCommand 메서드는 호출 대상 UI를 초기화한 후, 대상 UI의 ExecuteCommand 메서드를 호출한다. 따라서 호출 대상 UI를 로드한 후, 호출 대상 UI에서 특정 동작을 해야 할 경우에는 ExecuteCommand 메서드가 작성되어 있어야 한다.

[Fig5.2.1]은 ExecuteCommand 메서드가 작성되어 있는 예제이다. 예제 내에서 ExecuteCommand 메서드는 매개변수로 전달 받은 Lot Name을 텍스트 박스에 할당하고, UI 내의 LoadDataAsync 메서드를 호출하여 Lot 정보를 로드한다.

```
public override void ExecuteCommand(ArgumentPack arguments)
{
    #region Execute Command

    try
    {
        this.txtLotName.Text = arguments[BIZModel._COLUMN_NAME_LOT].ArgumentString;
        this.LoadDataAsync();

        return;
    }
    catch (System.Exception ex)
    {
        throw ex;
    }

    #endregion
}
```

[Fig5.2.1]

## 6. 복수의 매개변수 전달하기

TAP FX에서는 복수의 매개변수를 전달하기 위해 ArgumentPack을 사용한다. ArgumentPack은 복수의 Argument들의 컬렉션이다.

### 6.1. Argument

Argument 개체는 [Table6.1.1]과 같이 구성된다.

구성요소	설명	비고
Name	Argument Pack내에서 Argument의 유일성을 보장하는 이름	
Type	Argument Value의 타입	
Value	Argument의 Value	

[Table6.1]

### 6.2. ArgumentPack

앞에서 설명한 것과 마찬가지로 ArgumenPack은 복수의 Argument들의 컬렉션이다. 프로그래밍시에는 ArgumentPack의 Instance를 생성한 후, AddArgument 메서드를 호출하여 ArgumentPack에 Argument Instance를 추가하는 방법을 권장한다.

[Fig6.2.1]은 ArgumentPack의 Instance를 생성하고, "LOTNAME"이라는 Argument를 추가한다.

```
tmp = new ArgumentPack();
tmp.AddArgument("LOTNAME", typeof(string), tmpLotName);
```

[Fig6.2.1]

### 6.3. 매개변수로 전달 받은 ArgumentPack 사용하기

ArgumentPack은 Key-Value 형태로 구성되어 있기 때문에, Key를 사용하여 ArgumentPack의 특정 Argument

에 대한 접근이 가능하다. [Fig6.3.1]은 ArgumentPack의 특정 Argument에 접근하여, String 타입의 값을 가져온다.

```
this.txtLotName.Text = arguments[BIZModel._COLUMN_NAME_LOT].ArgumentString;
```

[Fig6.3.1]

Argument의 값이 String이 아닌 경우 ArgumentValue 프로퍼티를 사용하여 값을 가져오며, 이 때 형변환 코드를 사용해야 한다. [Code6.3.1]은 ArgumentValue 프로퍼티를 사용하여 Argument의 값을 가져오는 몇 가지 예제이다.

```
int number1 = (int)arguments["NUMBER"].ArgumentValue;
List<string> list1 = (List<string>)arguments["LIST"].ArgumentValue;
DateTime time1 = (DateTime)arguments["TIME"].ArgumentValue;
DataSet ds1 = (DataSet)arguments["DATASET"].ArgumentValue;
Lot ds1 = (Lot)arguments["LOTINFO"].ArgumentValue;
```

[Code6.3.1]

## 7. TAP BIZ Controls

### 7.1. Wafer Map Control

#### 7.1.1. Properties and Methods

Wafer Map Control은 UI에 Wafer내 각 DIE의 특성을 표시하기 위해 사용된다. Wafer Map Control이 지원하는 주요 Property는 [Table 7.1.1.1]과 같다.

Property	설명	비고
MapType	BINMAP, CD/THK Map 등의 구분 Default: Value: BINMAP	
MapStyle	Square Map 또는 Circle Map 구분 Wafer Map일 경우: Circle Map	
StartPointX	Map 좌측 시작 좌표값 Default Value: 11	
StratPointY	Map 상단 시작 좌표값: Default Value: 11	
MapData	Map Data 값을 가지는 DataTable	
ShowDummyCell	DIE좌표 Data외 Dummy Cell 표시여부 설정	
ShowShot	Shot표시 여부 설정	
ShowOuterLine	Map 외곽 테두리 표시여부 설정	
ShowTitle	Map Title 표시여부 설정	
Title	Map Title	
FlatZone	Map 기준방향(T/L/R/B)	
MarginWafer	Map 표시기준점까지의 여백의 값	

[Table7.1.1.1]

Property	설명	비고
DieXColumn	MapData에서 Die X 좌표값의 Column 이름	
DieYColumn	MapData에서 Die Y 좌표값의 Column 이름	
PointXColumn	MapData에서 Point X 좌표값의 Column 이름	
PointYColumn	MapData에서 Point Y 좌표값의 Column 이름	

[Table7.1.1.1]

Wafer Map Control이 지원하는 주요 Method는 [Table7.1.2]와 같다.

Method	설명	비고
<b>CreateMap(DataTable dt)</b>	Map에 데이터를 바인드한다. dt: Map Data에 할당되는 DataTable	
<b>SetShot( int cellCntX, int cellCntY, Color color, float width)</b>	Shot을 설정한다. cellCntX: Shot X좌표 cellCntY: Shot Y좌표 color: Shot 테두리 색상 width: Shot 테두리 색/굵기	

[Table7.1.1.2]

### 7.1.2. Data Scheme

[Table7.1.2.1]은 Map 기준정보 Table Scheme이다.

Column	PK	Type	Description
FAB	√	VARCHAR2(40)	
TECH	√	VARCHAR2(40)	
LOT_CD	√	VARCHAR2(40)	
PROD	√	VARCHAR2(40)	
AREA_CD	√	VARCHAR2(40)	
OPER	√	VARCHAR2(40)	
DIE_X		NUMBER	DIE의 가로 개수
DIE_Y		NUMBER	DIE의 세로 개수
MAP_X		NUMBER	Map의 가로 크기
MAP_Y		NUMBER	Map의 세로 크기
FLAT_ZONE		VARCHAR2(40)	
SHOT_X		NUMBER	Shot의 가로 DIE 개수
SHOT_Y		NUMBER	Shot의 세로 DIE 개수
UPDATE_DT_TM		VARCHAR2(40)	
MAP_TYPE		VARCHAR2(40)	

[Table7.1.2.1]

[Table7.1.2.2]는 BIN Map용 Map Data Table Scheme이다.

Column	PK	Type	Description
FAB	√	VARCHAR2(40)	
TECH	√	VARCHAR2(40)	
LOT_CD	√	VARCHAR2(40)	
PROD	√	VARCHAR2(40)	
AREA_CD	√	VARCHAR2(40)	
OPER	√	VARCHAR2(40)	
DIE_X		NUMBER	DIE의 X 좌표
DIE_Y		NUMBER	DIE의 Y 좌표
MAP_VAL		VARCHAR2(40)	
UPDATE_DT_TM		VARCHAR2(40)	

[Table7.1.2.1]

[Table7.1.2.2]는 THK, CD Map Data Table Scheme스키마이다.

Column	PK	Type	Description
FAB	√	VARCHAR2(40)	
TECH	√	VARCHAR2(40)	
LOT_CD	√	VARCHAR2(40)	
PROD	√	VARCHAR2(40)	
AREA_CD	√	VARCHAR2(40)	
OPER	√	VARCHAR2(40)	
DIE_X		NUMBER	DIE의 X 좌표
DIE_Y		NUMBER	DIE의 Y 좌표
POINT_X		NUMBER	DIE내 X 좌표위치
POINT_Y		NUMBER	DIE내 Y 좌표 위치
MAP_VAL		VARCHAR2(40)	
UPDATE_DT_TM		VARCHAR2(40)	

[Table7.1.2.2]

## 8. Namespace Naming Rules

### 8.1. 개요

Namespace는 아래의 명명규칙에 따른다.

#### Prefix.System Name.Main Menu(Service Group)

각 단위 기능(Service 또는 UI)는 위에서 정의한 Namespace 뒤에 명명한다. 각 개발환경의 공용/공통/상위 Namespace일 경우, "Main Menu (Service Group)를 생략할 수 있다.

### 8.2. Prefix

Prefix는 [Table8.2-1]에 열거된 Prefix 중 한 개를 개발환경에 맞춰 선택하여 사용한다.

Prefix	Description	Remark
TAP	TAP Framework Namespace	TAP.Base.Database
IntegratedSystems.	Package 제품용 Namespace	IntegratedSystems.ISEM.Equipment
ISTS	Technical Service용 Namespace	ISTS.HWQQSEIS.Common

[Table8.2.-2]

### 8.3. System Name

#### 8.3.1. Package 제품

Package 제품의 경우, 해당 제품의 이름을 사용한다 (예: ISEM, ISM).

#### 8.3.2. Technical Service

Technical Service의 경우, 고객사코드+시스템 조합을 사용한다 (예: HWQQNEIS). 단, Technical Service의 경우, 고객사에서 요청하는 이름을 사용할 수 있다.



## 8.4. AssemblyInfo

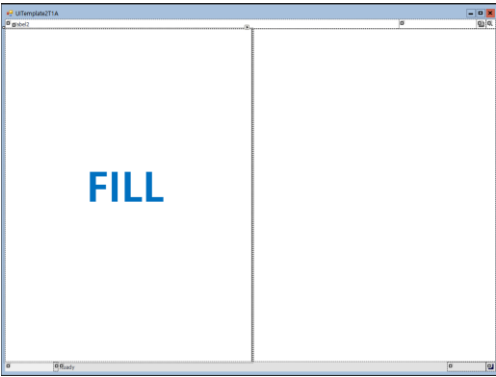
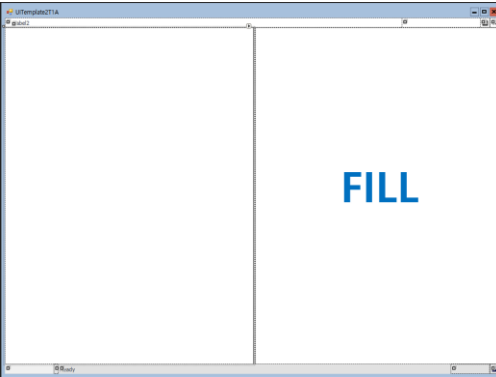
AssemblyInfo의 각 항목은 [Table8.4-1]과 같이 작성한다.

Item	Description	Remark
<b>Assembly Title</b>	해당 DLL의 이름 또는 Main Namespace	TAP.Base.Database
<b>Assembly Description</b>	-	
<b>Assembly Configuration</b>	-	
<b>Assembly Company</b>	ISSET-DA Shanghai Co., Ltd.	
<b>Assembly Product</b>	Package: Package 이름.	ISM, ISEM
	Technical Service: 경우 Project 이름	
<b>Assembly Copyright</b>	Package: ©iSET-DA Shanghai 2011-2022	개발시작연도~현재연도
	Technical Service: 프로젝트 환경에 따름	
<b>Assembly Trademark</b>	-	
<b>Assembly Information Version</b>	x.x.x.x의 형태로 관리하기 편하도록 작성	

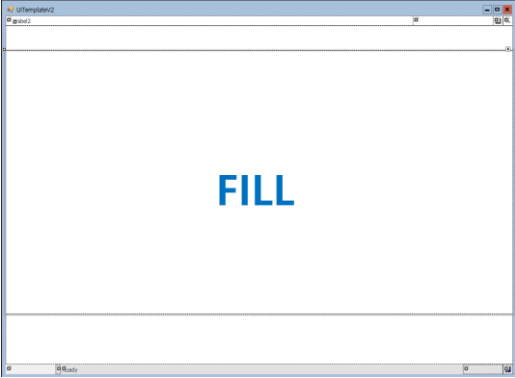
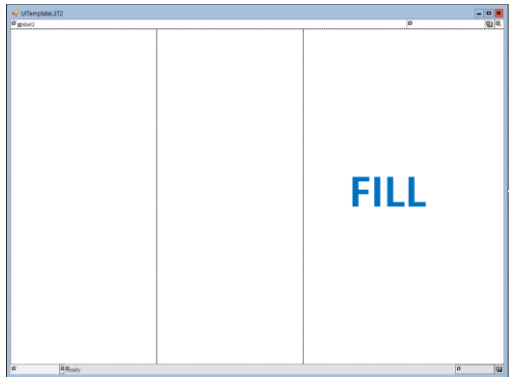
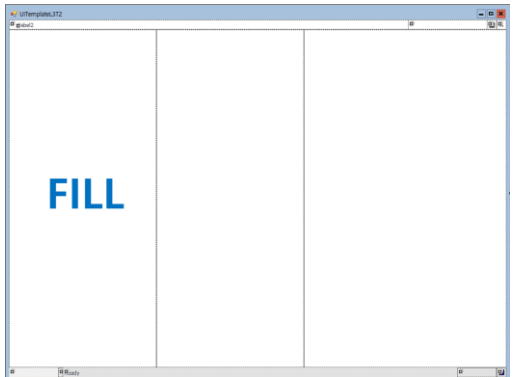
[Table8.2.-2]

# Appendix 1. UI Templates

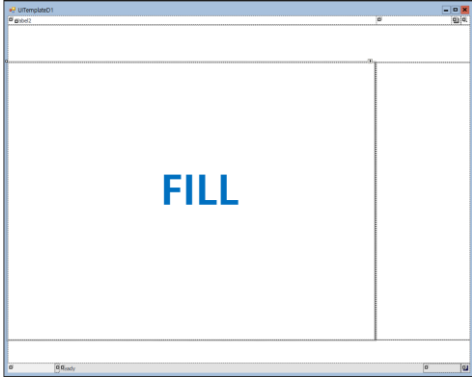
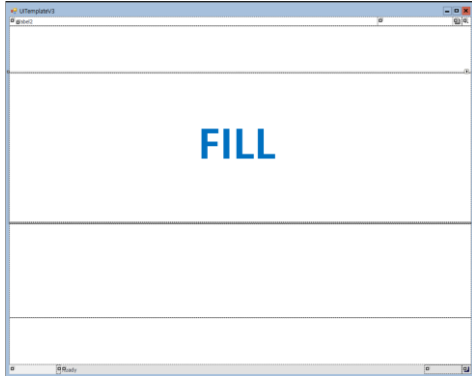
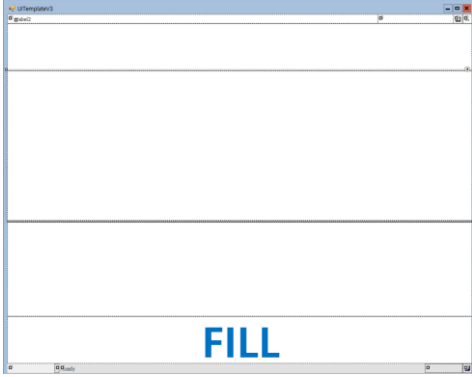
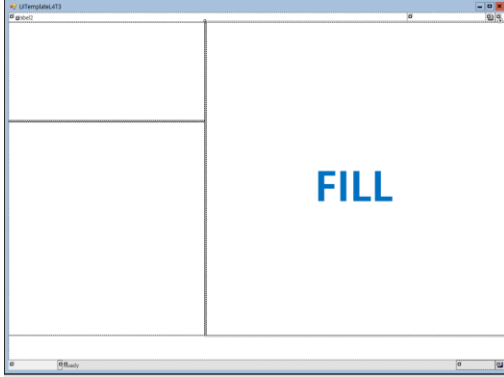
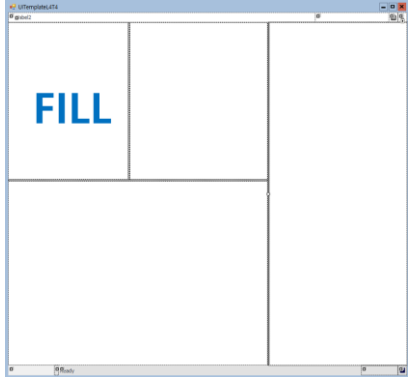
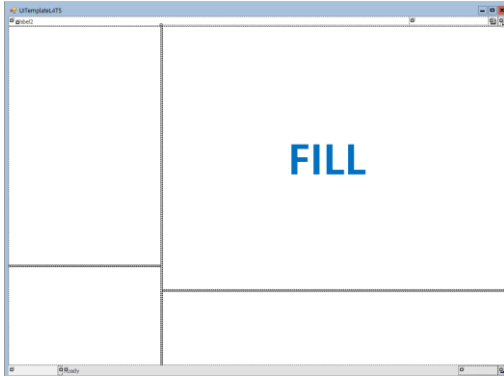
## Appendix 1.1 2Layer Templates

UI Template 2T1A	UI Template 2T1B
	

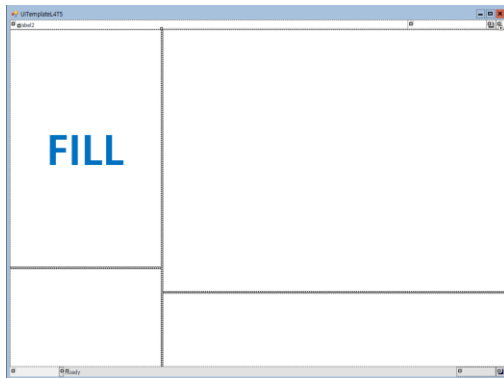
## Appendix 1.2 3Layer Templates

UI Template 3T1	
	
UI Template 3T2	UI Template 3T2A
	

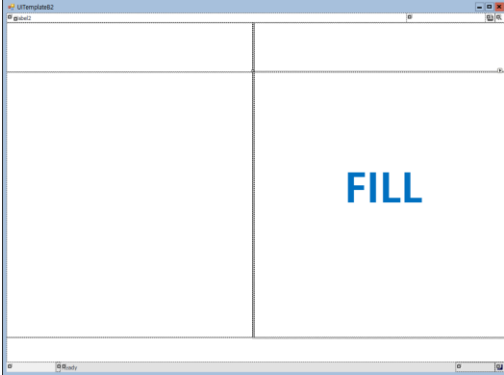
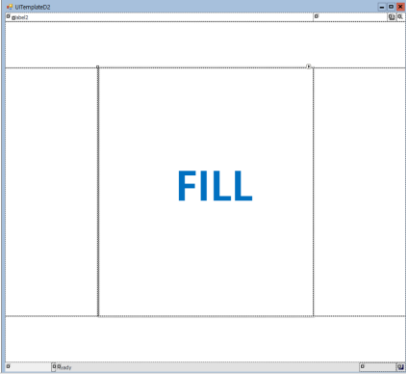
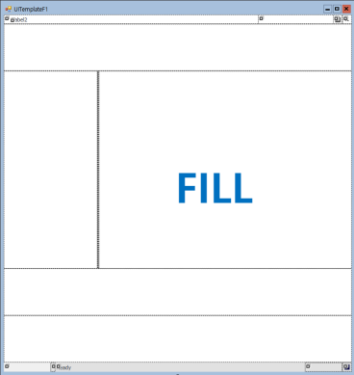
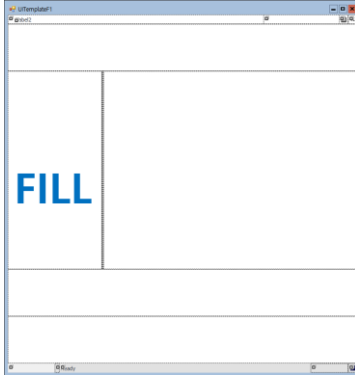
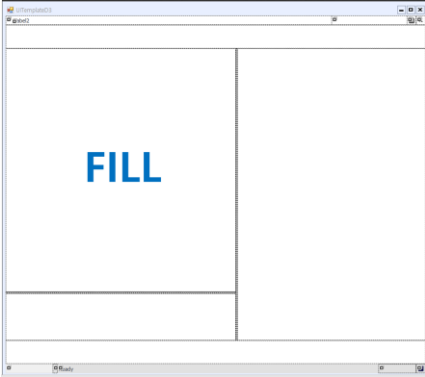
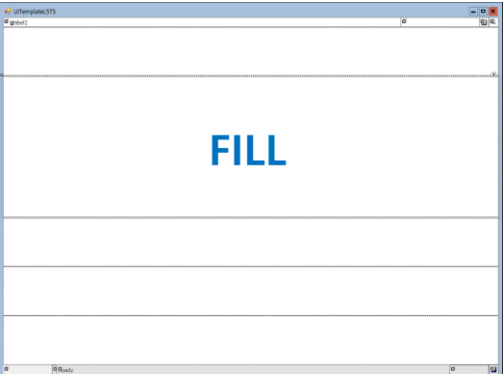
## Appendix 1.3 4Layer Templates

UI Template 4T1	UI Template 4T2
	
UI Template 4T2A	UI Template 4T2A
	
UI Template 4T4	UI Template 4T5
	

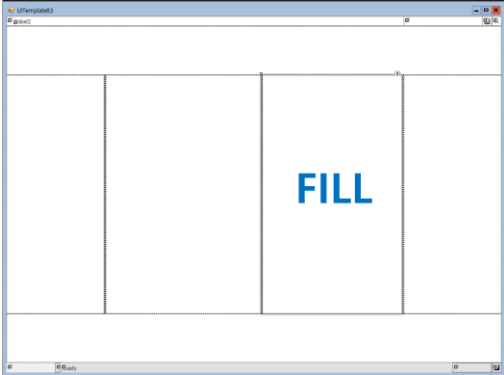
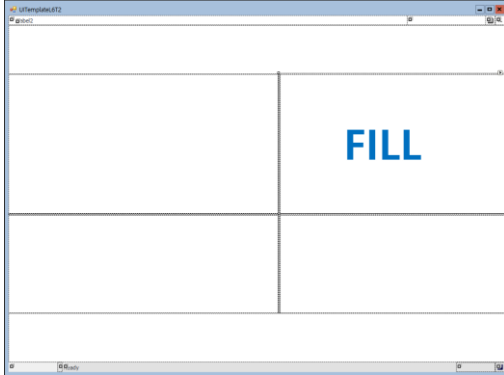
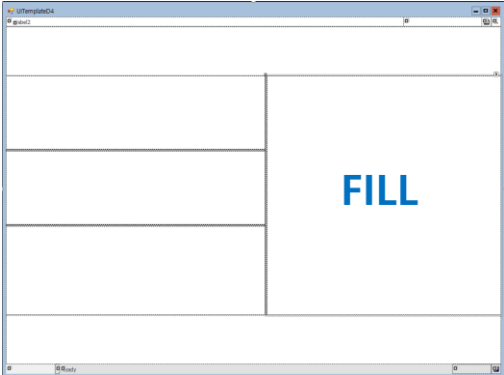
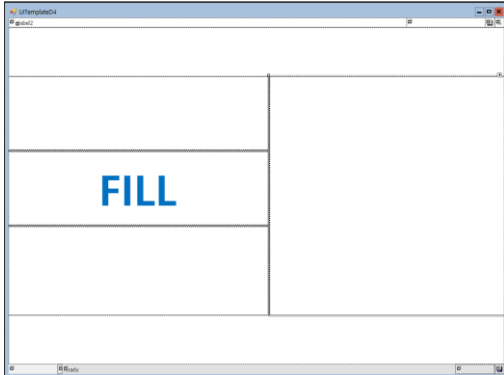
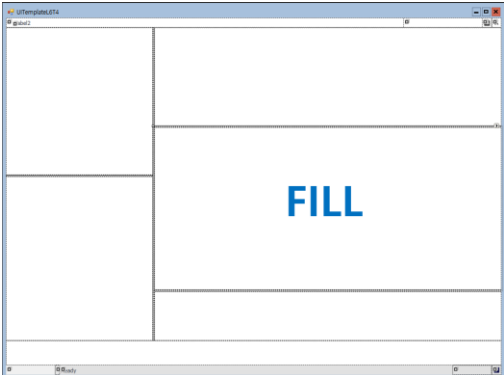
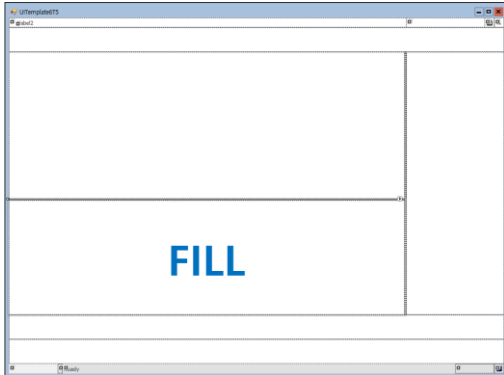
## UI Template 4T5A



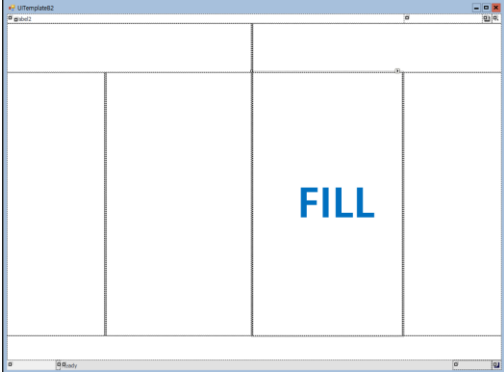
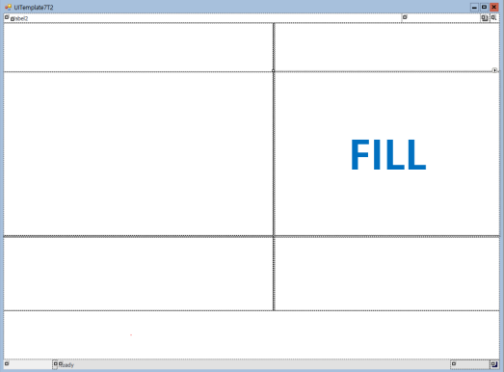
## Appendix 1.4 5Layer Templates

UI Template 5T1	UI Template 5T2
	
UI Template 5T3	UI Template 5T3L
	
UI Template 5T4	UI Template 5T5
	

## Appendix 1.5 6Layer Templates

UI Template 6T1	UI Template 6T2
	
UI Template 6T3	UI Template 6T3L
	
UI Template 6T4	UI Template 5T5
	

## Appendix 1.6 7Layer Templates

UI Template 7T1	UI Template 7T2
	



## Appendix 2. 표준 코드작성방법

### Appendix 2.1. 공통사항

소스 코드를 작성할 때 다음의 사항들을 준수해야 한다.

- 상수를 제외한 Namespace, Class 및 모든 Member들의 이름은 Camel Case를 사용한다.
- 모든 Public Member에는 XML 형식 주석을 추가하고, 영어로 간단한 설명을 작성한다.

### Appendix 2.2 멤버 종류별 배치 순서

Importing 목록의 경우, [FigA2.2-1]과 같이 상단에 .NET Framework 참조 Namespace를 나열하고, 그 아래에 TAP FX 참조 Namespace를 나열한다.

Class 내부에는 [FigA2.2-1]과 같이 Member 종류별로 #region 지시자를 사용하여 묶는다. 멤버 종류별 배치 순서는 다음과 같다.

- Constant
- Field
- Property
- Create and Disposer
- Initialize
- Method
- Event Handlers

```

1  using System;
2      using System.Collections.Generic;
3      using System.ComponentModel;
4      using System.Data;
5      using System.Drawing;
6      using System.Linq;
7      using System.Text;
8      using System.Windows.Forms;
9
10     using TAP.Base;
11     using TAP.Models;
12     using TAP.Models.SystemBasic;
13     using TAP.Models.Systems.Servers;
14     using TAP.UI;
15
16     namespace TAP.App.Lode.History
17     {
18         3 references
19         public partial class FormDiskHistory : TAP.UI.UITemplate5.UITemplateL5T4
20         {
21             Constants
22
23             Fields
24
25             Properties
26
27             Creator and Disposer
28
29             Initialize
30
31             Load Data
32
33             View Detail And Edit
34
35             Excel Exports
36
37             Event Handlers
38         }
39     }

```

[FigA2.2-1]

## Appendix 2.3 Constant, Field 및 Property 선언

### Appendix 2.3.1. Constant 선언

Constant를 선언할 때는 특별한 경우를 제외하고 public 한정자를 사용한다. Constant의 Naming Rule은 다음을 따른다.

- 기본적으로 모두 대문자를 사용한다.
- 필요할 경우 이름 중간에 "\_"를 사용한다.
- Constant의 이름은 "\_"로 시작한다.

```
/// <summary>
/// Argument key of box status
/// </summary>
public const string _ARGUMENTKEY_BOXSTATUS = "A_BOXSTATUS";
```

[Fig A2.3.1-1]

### Appendix 2.3.2. Field 선언

Field를 선언할 때는 특별한 경우를 제외하고 private 한정자를 사용한다. Field의 Naming Rule은 다음을 따른다.

- Field의 이름은 "\_"로 시작한다.
- Field 이름의 첫번째 글자는 소문자를 사용한다.

```
#region Fields

private string _user;
SystemDefaultInfo _defaultInfo;
List<Model> _historyData;

private new string _excelFilePath;

#endregion
```

[Fig A2.2-1]

### Appendix 2.3.3. Property 선언

Property를 선언할 때는 특별한 경우를 제외하고 public 한정자를 사용한다. Property의 Naming Rule은 다음을 따른다.

- Property 이름의 첫번째 글자는 “대문자”를 사용한다.

```
/// <summary>
/// Disk drive name that os starts
/// </summary>
0 references
public string BootDevice{get{return this.bootDevice;}set{this.bootDevice=value;}}
```

[Fig A2.2-2]

또한 모든 public 및 protected 멤버에는 XML 주석을 추가하고, 주석 내용은 영어로 작성한다.

## Appendix 2.4. Creator와 Disposer 작성

### Appendix 2.4.1. Creator 작성

Creator의 내용은 [Fig A2.4.1-1]과 같이 #region 처리기를 통해 묶는다. 또한, Creator가 “InitializeComponent” 호출 외의 동작을 할 경우에는 [Fig A2.4.1-2]와 같이 try-catch문을 사용한다.

```
/// <summary>
/// This creator creates instance of this.
/// </summary>
/// <param name="name">Model Name</param>
1 reference
public BoxStockModel(string name)
{
    Creator
}
```

[Fig A2.4.1-1]

```

/// <summary>
/// This creator creates instance of this.
/// </summary>
/// <param name="name">Model Name</param>
1reference
public BoxStockModel(string name)
{
    #region Creator

    try
    {
        this.CreateInstance(name);
    }
    catch(System.Exception ex)
    {
        throw new TAPModelException(MethodInfo.GetCurrentMethod(), e
    }

    #endregion
}

```

[Fig A2.4.1-2]

Creator내에 지역변수를 사용할 경우, 지역변수는 Code의 첫 시작부분(try-catch 구문 밖)에 선언한다.

## Appendix 2.4.2 Disposer 작성

.NET Framework의 특성상 Deposer를 반드시 작성할 필요는 없지만, Disposer를 작성해야 할 경우 다음의 Rule을 따른다.

- Disposer의 내용은 #region 지시기를 사용하여 묶는다([Fig A2.4.2-1]).
- Disposer의 내용은 try-catch 구문을 사용한다.
- catch 구문에서 별도의 Exception을 발생시키지 않도록 한다(필요할 경우 로그를 남기도록 한다).

```

/// <summary>
/// This method disposes this model.
/// </summary>
637 references
public override void Dispose()
{
    Dispose
}

```

[Fig A2.4.2-1]

```

/// <summary>
/// This method disposes this model.
/// </summary>
637 references
public override void Dispose()
{
    #region Dispose

    try
    {
        this.capacity = 0;
        this.realQuantity = 0;

        this.productDistinction = null;
        this.packingUser = null;
        this.packingShift = null;
        this.qraCheck = null;
        this.checkINDocumentNo = null;

        this.boxStatus = EnumStockStockInOutStatus.WAIT;
        //this.domesticExport = EnumStockDomesticExport.EXPORT;

        this.products.Dispose();

        base.Dispose();
    }
    catch
    {
        //
    }

    #endregion
}

```

[Fig A2.4.2-2]

## Appendix 2.5. Initialize 메서드 및 메서드 작성

메서드를 작성할 때, 다음의 사항을 준수한다. 메서드의 작성 예는 [Fig A2.5-1]을 참조한다.

- 메서드의 이름은 반드시 대문자로 시작한다.
- 전체 코드의 내용을 #region 처리기로 묶는다.
- 전체 로직 코드는 try-catch 문으로 묶는다.

- 지역변수는 메서드의 시작 부분에서 선언하며, try-catch 구문 밖에서 선언한다.
- 클래스를 초기화하는 메서드의 경우 Initialize라는 이름을 사용한다.

```
private void ViewDetail(int selectedRow)
{
    #region View Detail

    string tmpName = string.Empty;
    Model model = null;

    try
    {
        if (selectedRow < 0)
            return;

        tmpName = this.tSheet1.GetString(this.tSheet1.KeyColumn, this

        foreach (Model tmpModel in this._historyData)
        {
            if (tmpModel.Name.Equals(tmpName))
            {
                model = tmpModel;
                break;
            }
        }

        if (!object.Equals(model, null))
            this.controlModelView1._Model = model;

        return;
    }
    catch (System.Exception ex)
    {
        throw ex;
    }

    #endregion
}
```

[Fig A2.5-1]

## Appendix 2.6 Event Handler 작성

Event Handler를 작성할 때, 다음의 사항을 준수한다. Event Handler 작성 예는 [Fig A2.6-1]을 참조한다.

- Event Handler에는 실제 동작을 수행하는 메서드를 호출하는 수준으로 간단하게 작성한다.

- 전체 로직은 try-catch문으로 묶는다.
- catch문에는 구체적인 예외처리 코드를 작성한다(Popup, Logging 등)

```
1reference
private void FormServerHistory_Load(object sender, EventArgs e)
{
    #region FormServerModeler_Load

    try
    {
        this.Initialize();
    }
    catch (System.Exception ex)
    {
        string tmpMsg = _translator.ConvertGeneralTemplate(Fmessage.E
        TAPMsgBox.Instance.ShowMessage(this.Text, EnumMsgType.ERROR,
    }

    #endregion
}
```

[Fig 2.6-1]

## Appendix 2.7 Control Naming

Control의 이름은 Control의 종류를 의미하는 Prefix와 Control의 역할을 의미하는 이름의 조합으로 작성한다.

예를 들어 사용자의 이름을 입력해야 하는 Textbox Control이라면, Textbox를 의미하는 Prefix "txt"와 해당 텍스트 박스의 역할을 의미하는 "UserName"을 조합하여 "txtUserName"과 같은 형식으로 작성한다.

Control의 종류를 나타내는 Prefix는 모두 소문자를 사용하고, Control의 역할을 나타내는 이름은 대문자로 시작하는 Camel Case를 사용한다.

[Table A2.7-1]은 주요 Control의 Prefix이다.



.NET Control	TAP Control	Prefix	예
Button	TButton	btn	btnRun
CheckBox	TCheckBox	chk	chkViewOnly
CheckedListBox	TCheckedListBox	clb	clbCheckList
Panel	TCollapsiblePanel, TPanel	pnl	pnlRight
ComboBox	TComboBox	cmb	cmbYear
ContextMenu	TContextMenu	cxm	cxmContextMenu
DateTimePicker	TDateTimePicker	dtp	dtpStartTime
	TDateTimePickerSE	dts	dtsTerm
GroupBox	TGroupBox, TTitledGroupBox	gpb	gpbUserInfo
Label	TIconLabel, Label	lbl	lblFacility
ListBox	TListBox	lbx	lbxLotList
ListView	TListView	lvw	lvwLotList
	TNumericBox	nxt	nxtTimes
PictureBox	TPictureBox	pbx	pbxDefectImage
ProgressBar	TProgressBar, TSolidPrgoressBar	prb	prbProgress
RadioButton	TRadioButton	rbt	rbtMode
Splitter	TSplitter	spt	sptAB
TabControl	TTabControl	tab	tabRight
TextBox	TTextBox	txt	txtUserName
Treeview	TTreeView	tre	treDirectory

[Table A2.7-1]

## Appendix 3. 설치 문제점 및 해결방법

### Appendix 3.1. 로그인 창에 Facility 정보가 표시되지 않는 경우 또는 Log In 창 표시와 동시에 System.Null.Exception이 발생하는 경우

Database 연결 오류가 발생하였을 때, 해당 현상이 발생한다.

Database 연결시 오류가 발생할 경우, 첨부된 파일 "odp.net4.zip"의 압축을 해제하고, 압축 해제된 폴더내의 폴더 (bin, odp.net) 폴더를 오라클 설치 폴더에 X-Copy 한다.

### Appendix 3.2. 서비스 설치후 서비스가 정상적으로 구동되지 않는 경우

Database 연결 문자열이 올바르지 않을 때, 해당 현상이 발생한다.

올바른 데이터베이스 문자열을 다음의 Configuration 파일의 DB Section에 올바른 데이터베이스 문자열을 입력한다.

- TAPRemoting.Server.ConsoleHost.exe.config
- TAPRemoting.Server.NTHost.exe.config
- TAPRemoting.Server.Listener.exe.config

### Appendix 3.3. 다국어 로딩 또는 Access DB 연결시 오류가 발생하는 경우

이 경우 첨부된 파일 "AccessDatabaseEngine\_X64.exe"을 설치한 후, 프로그램을 다시 시작한다.

### Appendix 3.4. OracleManagedDataAccess 사용시 Database 연결이 되지 않는 경우

OracleManagedDataAccess 연동시 FX Configuration 파일 Database Section/Connection Element의 ConnectionString Key에 TNS Alias 대신 "모든 연결정보"를 입력한다 ([FigA3.4-1].

```
ConnectionString="Data Source=(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = isetsvr2)(PORT = 1521)) (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = ISEM))
```

[Fig3.4-1]

