

# Developer Guide

**Integrated Systems**

**Shanghai Headquarters**

**iSET-DA Shanghai**

**2019-2022**

# Contents

## 0. 概要

## 1. Client 安装

## 2. Framework 构成要素

## 3. User Interface 开发

## 4. Server-Client 通信

## 5. UI 之间通信

## 6. 传达复数的每个变数

## 7. TAP BIZ Controls

## 8. Namespace Naming Rule

## Appendix 1. UI Templates

## Appendix 2. 标准代码制作方式

## Appendix 3. 设置问题点及解决方法

## File Release

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

## 概要

该文件是为开发基于FAPFX和Integrated Systems的应用程序的开发者提供“开发标准”而编写的。本文包含 Application Setup, 客户端开发, 编码规则和Namespace命名规则等。

本文的Section 1和Section 2介绍了应用程序的基本配置, 这部分对于初级开发者可能有些难度。而且,正在阅读此文档的开发者是团队领导, 建议从Section 1开始阅读, 否则从Section 3开始阅读也没有问题。

正在阅读此文件的开发领导们, 请支持和指导组员们阅读相关文件, 并按照相关文件的标准进行开发。

# 1. Client 安装

## 1.1. Framework 设置以及 Directory 构成

把提供的 tapfat\_\*\_dev\_kit.zip 文件,在 Project的 Root Directory 进行压缩解除 ([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]

确认 FX Directory 的 \*.exe 文件, \*.exe.config 文件, \*.config 文件.

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]

在 [Fig1.1-3]의 \*.exe 文件是执行文件,\*.config 文件是 Application 的 Configuration 文件, \*.exe.config 文件是 Framework의 Configuration 文件.

## 1.2. Framework Configuration 构成

在 FX Directory 上,打开 \*.exe.config 文件.

### 1.2.1. Environment Section

构成 Environment Section. 这 Section 定义 Application의动作方式. Environment Section의各 Key의意义与

[Table1.2.1-1]一样.

Key	说明	备注
Region	在 Application 使用的最大的 Key值. 一般情况下 Application会分配设置地区的名字.	
InstallType	Application在 3Tier 环境上动作时, 分配 "CLIENT" . 若不是该情况时,分配 "SERVER" .	

[Table1.2.1-1]

变更 Region的 Key值时, 作为 Application Configuration 文件的 \*.Config的 Region Element 名字也许要变更.

### 1.2.2. Data Source Section

构成 Data Source Section. 该 Section 设定 Application的 Data Source. 如果 Application 不与 Data 联动时,也不需构成对应的 Section. Data Source Section的各 Key的意义与 [Table1.2.2-1] 一样.

Key	说明	备注
DataSource	以 Data Source 联动 Database时, 分配 "DB" . 联动 XML 文件时,分配 "XML" .	
InfoBase	与 Data Source 连接信息在 Framework Configuration设定时输入 "FX" . 如果不是该情况时输入 "APP" .	

[Table1.1.2-1]

### 1.2.3. Database Section

构成 Database Section. 设定该 Section与 Database的连接. 如果 Application与 Database 不通信或者在 3 Tier环境上动作时,对应的 Section 不需要构成. Database Section的各 Key的意义与 [Table1.2.3-1]一样.

Key	说明	备注
<b>Default</b>	输入基本联动的 Database 信息的Key. Application 联动单一的 Database时,分配 “ <b>DEFAULT</b> ” .	
<b>Timeout</b>	从 Database 分配最大的 Response时间(单位: 秒).	

[Table1.2.3-1]

根据在 Database Section上 Application 联动 Database 数量,添加 Connection Element. Connection Element的各 Key的意义与 [Table1.2.3-2]一样.

Key	说明	备注
<b>Key</b>	对应 Connection信息的 Key值. Application 联动单一的 Database时, 分配 “ <b>DEFAULT</b> ” .	
<b>DBMS</b>	分配联动的 DBMS的种类. 联动的 DBMS是 Oracle时分配 “ <b>ORACLE</b> ” ,SQL-SERVER时是 “ <b>MSSQL</b> ” ,DB2时是 “ <b>DB2</b> ” . 除此之外的 DBMS,当前版本不支持.	
<b>ConnectionString</b>	分配需连接的 Database的 Connection 信息.	
<b>SelectCommandMethod</b>	给 Database 传送 Query 命令时,分配要使用的方式. 以 Text形式传送命令时分配 “ <b>TEXT</b> ” , 以Stored Procedure形式传送命令时分配 “ <b>PACKAGE</b> ” .	
<b>ModifyCommandMethod</b>	给 Database 传送 Non-Query命令时,分配需使用的形式. 以 Text形式传送命令时分配 “ <b>TEXT</b> ” , 以Stored Procedure形式传送命令时分配 “ <b>PACKAGE</b> ” .	

[Table1.2.3-2]

#### 1.2.4. XML Section

构成 XML Section. 该 Section设定与 XML文件的连接. 如果 Application不联动 XML文件而是在 3Tier环境上运作时,对应的 Section 不需要构成. XML Section的各 Key的意义与 [Table1.2.4-1]一样.

Key	说明	备注
FileName	分配 XML文件的 Path.	
RootNodeName	分配 XML文件的 Root Node的名称.	
BackupPath	分配 XML文件的 Backup Path.	

[Table1.2.4-1]

### 1.2.5. Log Section

构成 Log Section. 该 Section设定 Application的 Logging. Log Section是以各 Log 类型依据的 Log Element所构成. Log Element的各 key的意义与 [Table1.2.5-1]一样.

Key	说明	备注
Type	Log的类型. 分配的字母表在 Sourcecode上,作成 Log相关的代码时使用.	
Logging	设定对应类型的 Log使用与否. 使用对应类型的 Log时分配 <b>"true"</b> , 不是时 分配 <b>"false"</b> .	
Path	对应类型的 Log文件储存的 Directory名称.	
Extension	对应类型的 Log文件所使用的扩展名.	
MaxSize	设定各 Log文件的大小(单位: Byte).	

[Table1.2.5-1]

### 1.2.6. Remote Listener Section

构成 Remote Listener Section. 该 Section设定与远程电脑所用的 Remote Service的通行. Application在 3Tier 环境下不运作时,该 Section不需要设定. Remote Listener Section的各 Key的意义与[Table1.2.6-1]一样.

Key	说明	备注
Multi Console	在远程电脑上有复数的 Remote Service运作时, 分配 <b>"true"</b> . 不是的话分配 <b>"false"</b> .	



[Table1.2.6-1]

Remote Host信息在 ListenerHost Element上设定. 该 Element的各 Key的意义与 [Table1.2.6-2]一样.

Key	说明	备注
IP	Remote Host运作的电脑的 IP地址	
Port	与 Remote Host通信所使用的 端口号	
MinRemoteCount	正在运作的 Remote Service的最低数量	
RetryCount	与 Remote Host通信失败时, 最多重试次数	
RetryCountInterval	与 Remote Host通信失败时, 到下一次重试的待时间(单位: MS)	

[Table1.2.6-2]

各 Remote Service信息是在 Listners Element内的 Listner Element上设定.需按 Remote Service的数量,添加 Listener Element.该 Element的各 Key的意义与 [Table1.2.6-3]一样.

Key	说明	备注
No	Remote Service的号码(固有值)	
Port	Remote Service在通信时所使用的端口号	

[Table1.2.6-3]

### 1.2.7. App Section

构成 App Section.该 Section设定 Application Launcher的构成. App Section的各 Key的意义与[Table1.2.7-1]一样.

Key	说明	备注
-----	----	----

<b>Facility</b>	在 Application需使用的第二大 Key值. 一般情况下分配设施的名称.	
<b>UserLanguage</b>	分配 Application运作时基本使用的语言. 韩国语时 “ <b>KR</b> ”, 英语时 “ <b>EN</b> ”, 中国语 “ <b>CN</b> ” . 除此以外的语言在当前版本上是不支持.	
<b>MDIDisplayName</b>	分配产品的名称. 在该 Key分配的名称以 Launcher的标题显示.	
<b>AppConfigFileName</b>	Application Configuration文件名称	
<b>AppConfigName</b>	Application Configuration文件的 Root Node名称	

[Table1.2.7-1]

#### 1.2.7.1. App Element

在 Apps Element添加 App Element 增加 Application Group. App Element的各 Key的意义与[Table1.2.7.1-1]一样.

Key	说明	备注
<b>Key</b>	Application Group的 Key值.	
<b>Sequence</b>	在 Application Launcher显示对应 Application Group的顺序.	

[Table1.2.7.1-1]

在 App Element添加 SubApp Element可以设定 Application信息. SubApp Element的各 Key的意义与[Table1.2.7.1-2]一样.

Key	说明	备注
<b>Key</b>	Application的 Key值.	
<b>AppDirectory</b>	对应 Application的 DLL文件的 Path.	

<b>Image</b>	分配在 Application Launcher显示的 Image的文件名称.	
<b>HoverImage</b>	分配在 Application的 Launcher上显示的 Hover Image的文件名称.	
<b>Argument</b>	分配 Application开始时所需的每个变数的值.	
<b>AppCfg</b>	分配 Application固有的 Configuration文件名称.	
<b>Enabled</b>	分配把 Application显示在 Launcher与否. Application显示在 Launcher时分配 <b>"true"</b> 不是时分配 <b>"false"</b> .	
<b>DisplayName</b>	分配在 Launcher和标题栏显示的 Application的名称.	

[Table1.2.7.1-2]

### 1.2.7.2. Startup UI Element

使用 Startup UI Element设定 Application开始时实行的 UI. Startup UI Element的各 Key的意义与 [Table1.2.7.2-1]一样.

Key	说明	备注
<b>Enabled</b>	Application开始时指定对特定界面的实行与否. 特定界面开始时分配 <b>"true"</b> 不是时分配 <b>"false"</b> .	
<b>StartUpMainMenu</b>	Application开始时对实行界面的主要菜单进行名称分配.	
<b>StartUpMenu</b>	Application开始时分配实行界面的菜单名称.	

[Table1.2.7.2-1]

### 1.2.8. Cross Language Section

构成 Cross Language Section.该 Section设定 Application多国语. Cross Language Section的各Key的意义与 [Table1.2.8-1]一样.

Key	说明	备注
<b>LocalFile</b>	分配多国语 MED文件的名称.	
<b>ServerDirectory</b>	分配在 Server上的多国语 MDB文件的 Sever Path.	

<b>NeedToDownload</b>	设定多国语 EDB文件从 Server是否需要下载. Application Launcher 实行时需要下载多国语 MDB文件时分配 <b>"true"</b> , 不是时分配 <b>"false"</b> .	
<b>NeedToApply</b>	设定是否使用多国语. 使用多国语时分配 <b>"true"</b> , 不是时分配 <b>"false"</b> .	

[Table1.2.8-1]

## 2. Framework 构成要素

TAP FX是以 Base, Data, Fressage, Mini, Model, Remoting, Service Base, UI, UI Control, Web, Workflow, App Base所构成. 各构成要素的作用与[Table2-1]相同.

构成要素	说明	备注
<b>Base</b>	共用 Library, Application所构成,提供通信等 Application开发的基本要素 .	
<b>Fressage</b>	提供多国语功能.	
<b>Mini</b>	提供与 File DB(MDB 等)的通信.	
<b>Model</b>	加工 Data提供 Business对象.	
<b>Remoting</b>	提供以 TCP/IP基础的远程电脑之间的通信.	
<b>Service Base</b>	提供 Windows Service Application开发的基本要素.	
<b>UI</b>	提供 UI开发的基本要素.	
<b>UI Control</b>	提供开发 UI时所需的基本 Control.	
<b>Web</b>	提供 Web Application开发所需的基本要素.	
<b>Workflow</b>	提供 Workflow功能.	
<b>App Base</b>	提供在 Base.Configuraition不提供的 Application固有的构成要素得以设定的功能.	

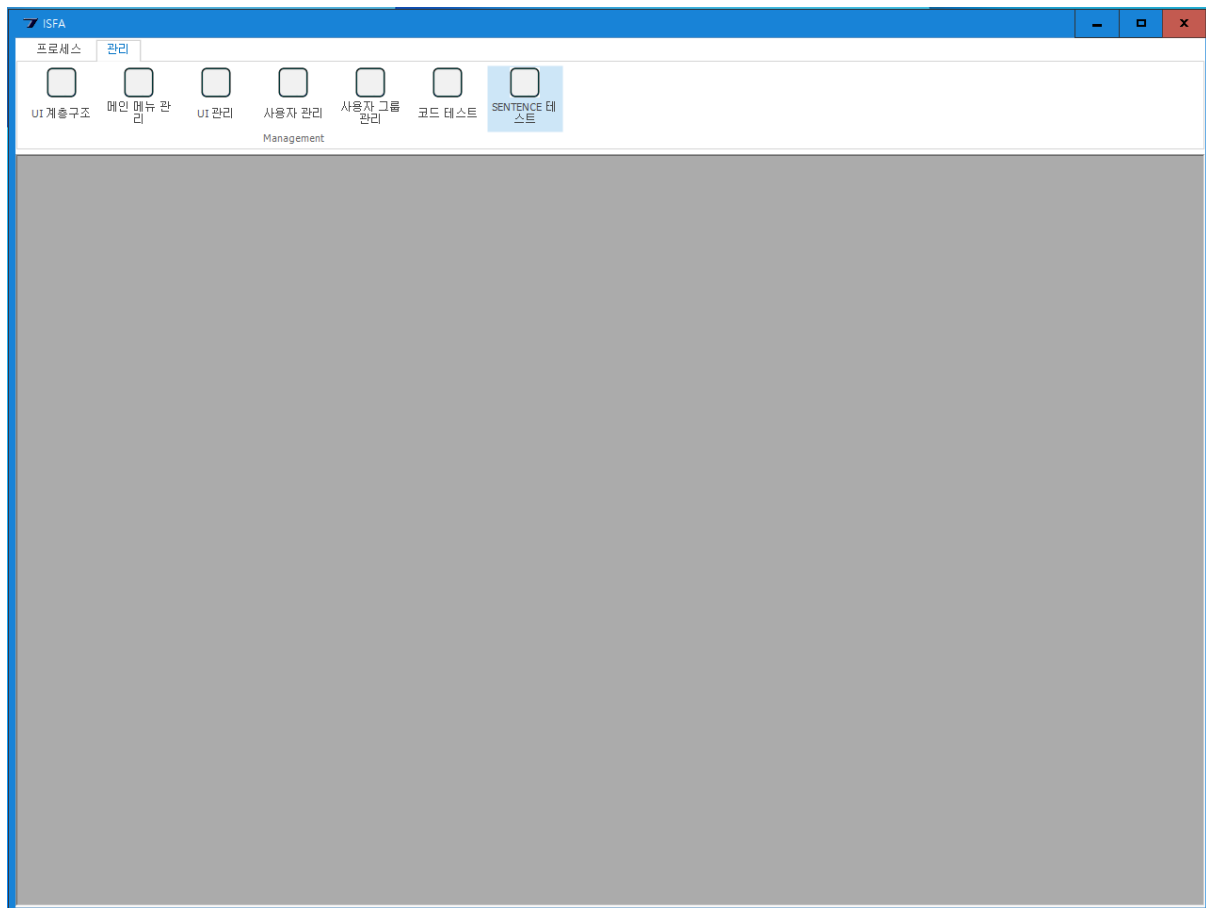
[Table2-1]

## 3. UI 开发

TAP FX 的 UI 拥有 Application MDI, Main Menu, UI 的分级结构.在该 Section 说明各分级结构的作用,说明 UI 开发所需的构成要素.

### 3.1. Application MDI

[Fig3.1-1]是 TAP FX for 的 MDI.



[Fig3.1-1]

## 3.2. 制作 UI

TAP FX环境的 UI是继承 TAP.UI.UITemplate制作. UI Template是 UI的 Layout. UI Template的种类和各 Layout需参照本文件的 Appendix 1.而且 UI Template继承 TAP.UI.UIBase. UI Base提供 UI制作所需的各种功能.

### 3.2.1. UI 制作准备.

为了在 TAP FX环境提供 UI, 需基本参照以下项目.

- 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. 在 Project添加新的 Windows Form.

3.2.1.3. 新的 Windows Form能够使用 UI Template需修改 Class的继承 Part. 该例子上做出 UI使用 UI Template L4T1([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. 为了新的 Windows Form的初始化, Initialize Method进行 Override. 进行 Override的 Initialize Method最优先呼叫 UI Base的 Initialize UI Method. UI Base的 InitializeUI Method呼叫句型下面,制作新的 Windows Form固有的初始化代码([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]

添加的 Initialize Method实行 UI时,是最先被呼叫的 Method, UI Base的 Initialize UI是进行支持各 Control的多国语言等基本初始化作业.

### 3.2.2. 使用 Callback 形式的非同步呼叫

TAP FX环境的 UI是提供 Callback呼叫功能.



3.2.2.1. 制作第一个需呼叫的 Method.该 Method必须具有返回值. [Fig3.2.2.1-1]以 List<Model>形式返还.

```
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]的代码是为了显示 Method具有返回值的例子.里面的代码当前理解不了也无妨.

3.2.2.2. 制作第二个呼叫的 Method(Callback 메서드). 该 Method是在 3.2.2.1制作的返回值形式拥有参数.

## 3.3. 多国语言设定

### 3.3.1. 多国语言设定文件

通过 TAP FX的构成要素的 Fessage Code Collection可以设定多国语. Fressage Code Collection是 MDB文件形式,位于 Framework Directory的 “mnls” Directory. 文件名称需与 Framework Configuration Cross Language Section의 LocalFile Key的值一致.

多国语设定文件的内部构造与 [Fig3.3.1-1]一样.

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]

各字段的意义与 [Table3.3.1-1]一样.

컬럼	说明	备注
<b>CODE</b>	Fressage Code. 制作 UI时, 使用该代码分配 Control의 Text值.	
<b>POS</b>	对应代码的词类. 在 Fressage所使用的词类参照 [Table3.3.1-2].	
<b>EN</b>	对应代码的英语单词(句)	
<b>KR</b>	对应代码的韩国语单词(句)	
<b>CN</b>	对代码的中国语单词(句)	
<b>CC</b>	对应单词的特性(代码的词类为 NOUN时才能分配)	

[Table3.1.1-1]

词类	说明	备注
<b>NOUN</b>	名词	
<b>ADJECTIVE</b>	形容词	
<b>VERB</b>	动词	
<b>ACRONYM</b>	保留词	没能以多国语转换.
<b>IDOM</b>	熟语. 以两个以上单词构成的代码	
<b>ADVERB</b>	副词	

[Table3.1.1-2]

### 3.3.2. 制作 Code Phrase(代码句)

Freemessage的基本 Converting单位是以一个以上的单词构成的 Code Phrase. 用户是以基础语法为依据构成 Code Phrase. [Table3.1.2-1]是 Code Phrase以各种语言 Converting的几种例子.

Code Phrase	英语	韩国语	中文	备注
<b>SEARCH</b>	Search	검색	调查	
<b>SEARCH RECIPE</b>	Search recipe	레시피 검색	调查配方	
<b>MOVE PITCH</b>	Move pitch	Pitch 이동	移动 Pitch	
<b>CURRENT CELL</b>	Current CELL	현재 CELL	当前 CELL	
<b>SCAN FIELD</b>	Scan field	필드 스캔	扫描 Field	

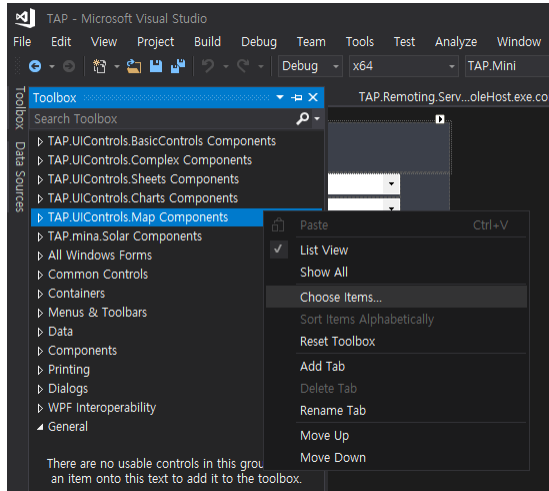
[Table3.1.2-1]

## 3.4. 使用 TAP UI Control

TAP FX是对能够更加便利使用 Visual Studio 所基本提供的 UI Control 或者 Third Party Control 进行 Wrapping的.

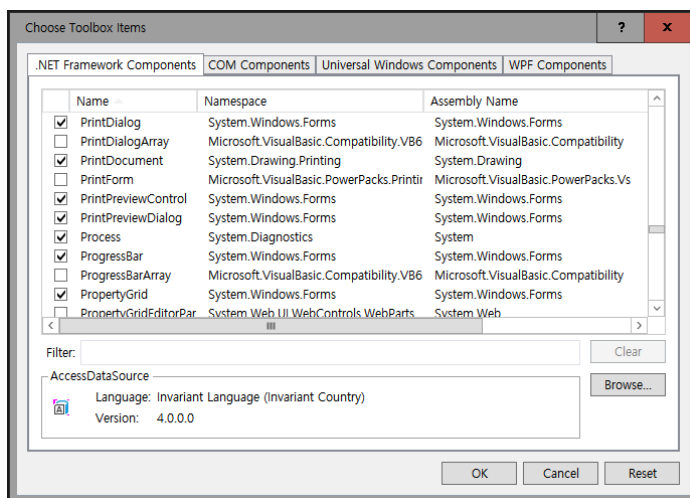
### 3.4.1. 在 Visual Studio 添加 TAP UI Control

3.4.1.1. 跟[Fig3.4.1.1-1]一样 在Visual Studio的 Toolbox 点击鼠标右键后, 在 Context Menu 选择[Choose Items...].



[Fig 3.4.1.1-1]

3.4.1.2. Choose Toolbox Items 窗口的初始化完成后, 点击[Browse]按钮.

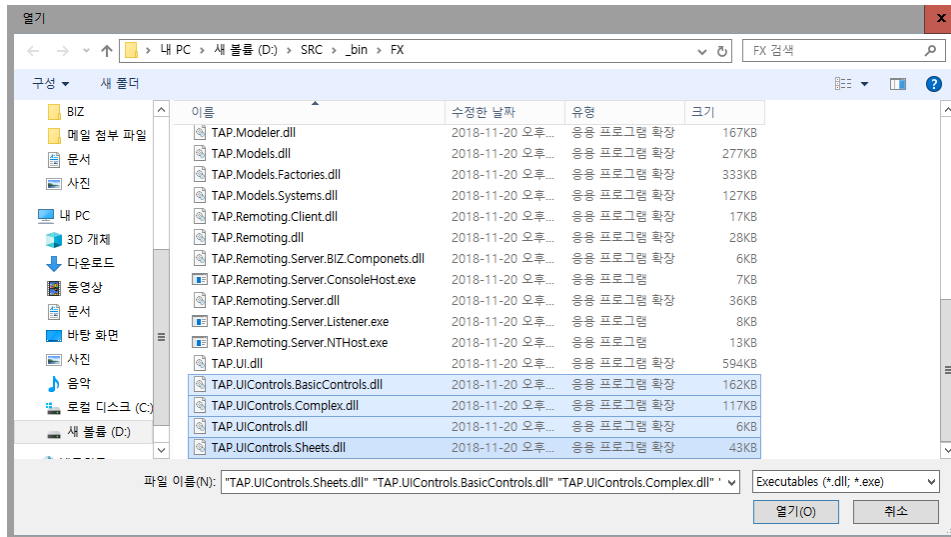


[Fig 3.4.1.2-1]

3.4.1.3. 在设置 TAP FX Client 文档上选怎如下 DLL 后, 点击[确认].

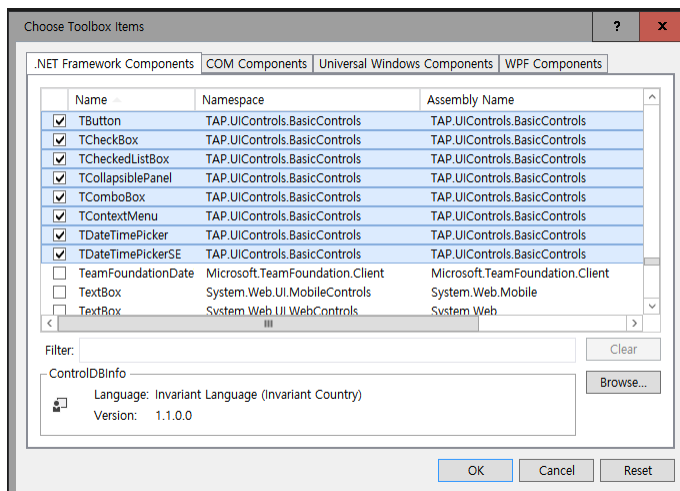
- TAP.UIControls.dll

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



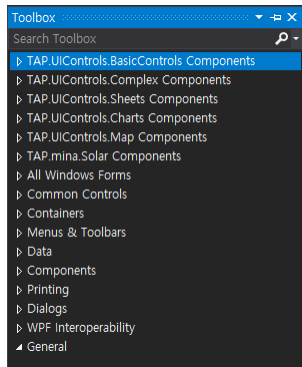
[Fig3.4.1.3-1]

3.4.1.4. 在 Choose Toolbox Items 窗口选择 TAP.UIConrols\*의 DLL 后, 点击[确认]键.



[Fig3.4.1.4-1]

3.4.1.5. 在 Toolbox 确认 TAP UI Control Load与否.

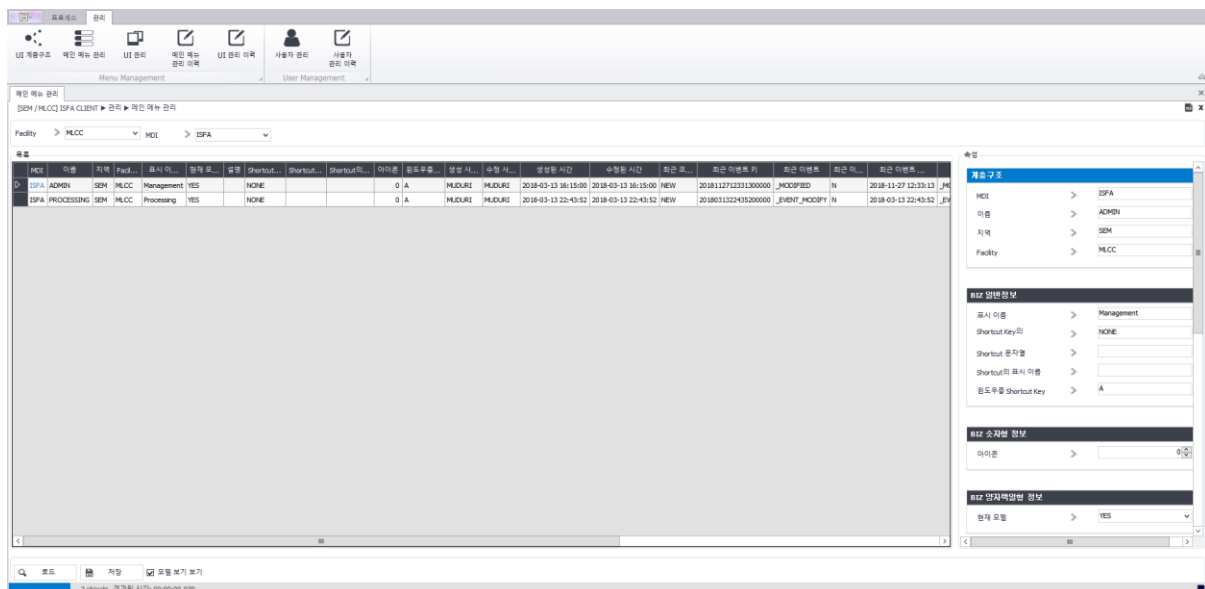


[Fig3.4.1.5-1]

## 3.5. 菜单管理

### 3.5.1. 主菜单管理

主菜单管理在[管理]-[主菜单管理]上进行([Fig3.5.1-1]).



[Fig3.5.1-1]

#### 3.5.1.1. 添加菜单

菜单的添加/删除/修改作业在菜单管理的界面的[属性]窗口进行.

### 3.5.1.1.1. 阶层构造

在[阶层构造]输入所属的对应菜单的 MDI名称,主菜单名称, Region, Facility ([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. 常规信息

在[常规信息]输入主菜单上的显示名称 (Display Name) ([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. 保存

所有信息输入完毕后, 点击界面左下侧的[保存]键进行保存.

### 3.5.1.2. 主菜单去除

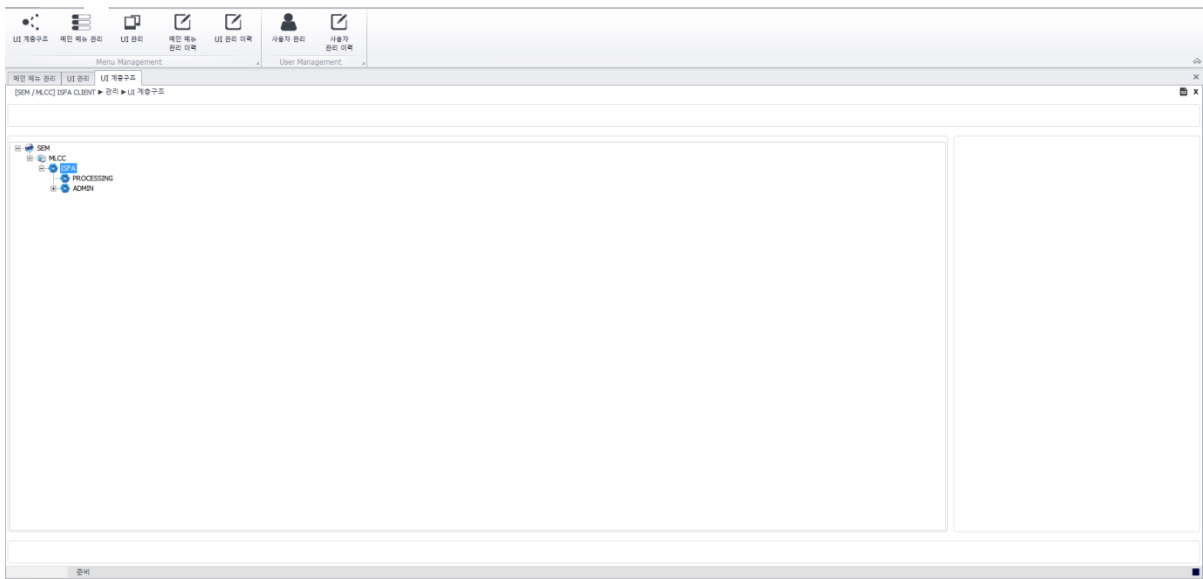
去除主菜单时,在[属性]的 [BIZ 两者择一型信息]上的[当前 Model]的值变更成 “NO” 后, 点击保存键 ([Fig3.5.1.2-1]).



[Fig3.5.1.2-1]

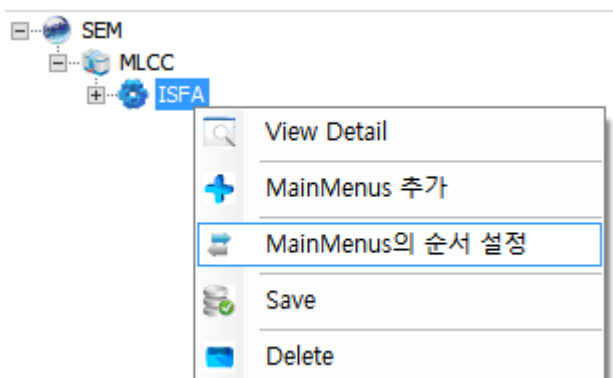
### 3.5.1.3. 主菜单顺序设定

主菜单顺序在[管理]-[UI阶层构造]上进行 ([Fig3.5.1.3-1]).



[Fig3.5.1.3-1]

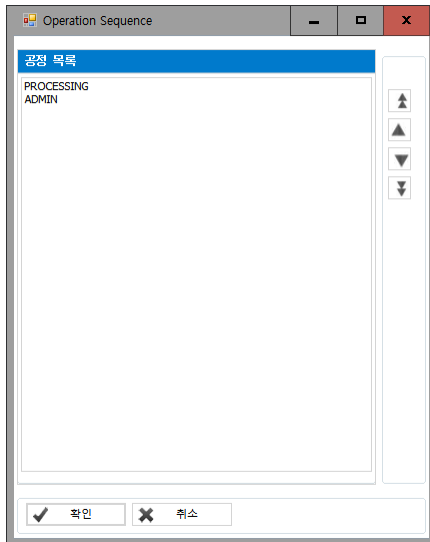
在树视图选择 MDI 名称后,单击鼠标右键选择对应显示的Context菜单上的 [MainMenus的顺序设定]  
([Fig3.5.1.3-2]).





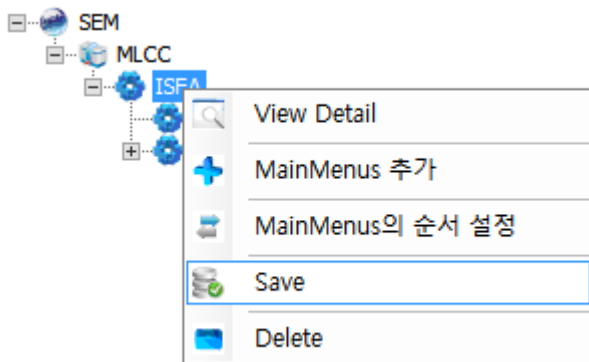
[Fig3.5.1.3-2]

[Fig3.5.1.3-3]一样,使用顺序设定窗口上的右侧箭头,设定顺序.



[Fig3.5.1.3-3]

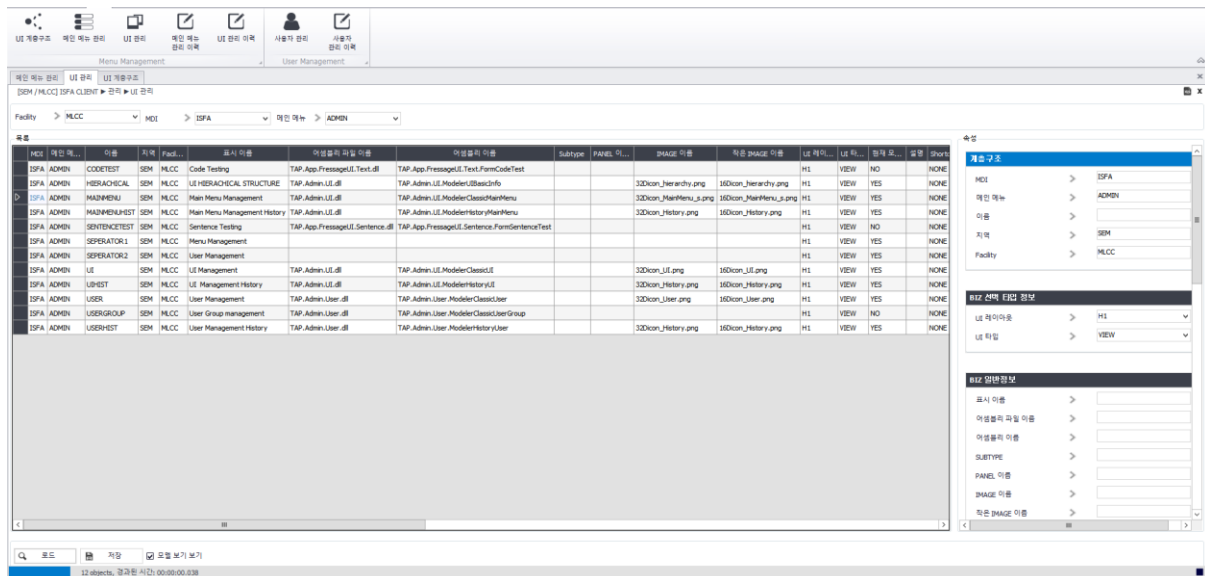
完成顺序设定, 选择 MDI 名称后, 点击鼠标键选择 Context 菜单上的 [Save]. ([Fig3.5.1.3-4])



[Fig3.5.1.3-4]

### 3.5.2. 菜单管理

主菜单管理在[管理]-[UI管理]上进行 ([Fig3.5.2-1]).



[Fig3.5.2-1]

### 3.5.2.1. 菜单添加

菜单的添加/删除/修改作业在 UI 管理 UI 右侧的 [属性]窗口进行.

#### 3.5.2.1.1. 阶层构造

在[阶层构造]输入所属的对应菜单的 MDI名称,主菜单名称, Region, Facility ([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. 常规信息

在常规信息输入的信息与 [Table3.5.2.1.2-1]一样.

名称	说明	备注
显示名称	UI的显示名称 (Display Name)	
Assembly 文件名称	包括 UI 的 Assembly 名称	需要包括扩展名输入
Assembly 名称	UI的 Assembly 名称	
IMAGE 名称	显示在菜单窗口的 Image 名称	对应的 Image 需在 Application 文件夹里的 [IMAGES]/[MENU]/[BIG] 文件夹里.
小 IMAGE 名称	在菜单窗口显示的小 Image 的名称	对应的 Image 需在 Application 文件夹里的 [IMAGES]/[MENU]/[SMALL] 文件夹里.

[Table3.5.2.1.2-1]

[Fig3.5.2.1.2-1]是构成常规信息的界面.

BIZ 일반정보

표시 이름

>

UI Management

어셈블리 파일 이름

>

TAP.Admin.UI.dll

어셈블리 이름

>

TAP.Admin.UI.ModelerClassicU

SUBTYPE

>

PANEL 이름

>

IMAGE 이름

>

32Dicon\_UI.png

작은 IMAGE 이름

>

16Dicon\_UI.png

Shortcut Key의

>

NONE

Shortcut 문자열

>

Shortcut의 표시 이름

>

윈도우줄 Shortcut Key

>

A

[Fig3.5.2.1.2-1]

### 3.5.2.1.3. 保存

所有信息输入完毕后, 点击界面左下侧的[保存]键进行保存.

### 3.5.2.2. 菜单去除

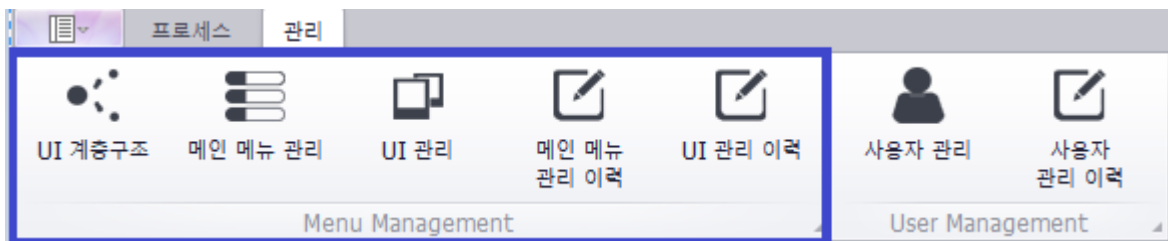
去除主菜单时,在[属性]的 [BIZ 两者择一型信息]上的[当前 Model]的值变更成 “NO” 后, 点击保存键 ([Fig3.5.2.2-1]).



[Fig3.5.2.2-1]

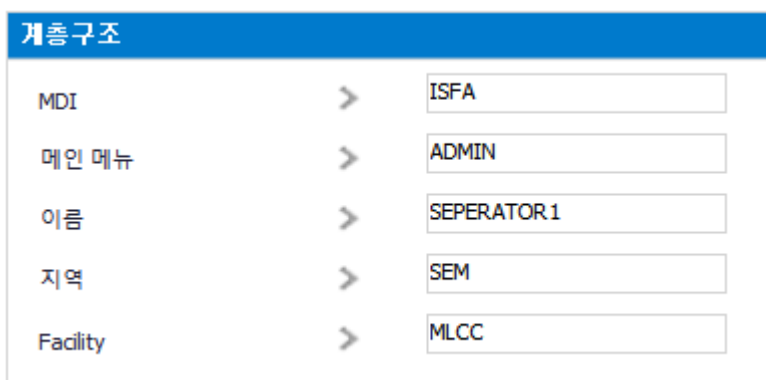
### 3.5.2.3. Page Group 设定

为了与 [Fig3.5.2.3-1]一样设定 Page Group 需添加 “SEPERATOR” .



[Fig3.5.2.3-1]

为了添加 “SEPERATOR” 需与 [Fig3.5.2.3-1 ]一样在 [阶层构造] 的 [名称]项目上输入 “SEPERATOR\*” .



[Fig3.5.2.3-2]

然后在 [常规信息] 项目的 [显示名称] 上输入对应的 Page Group 名称 ([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]

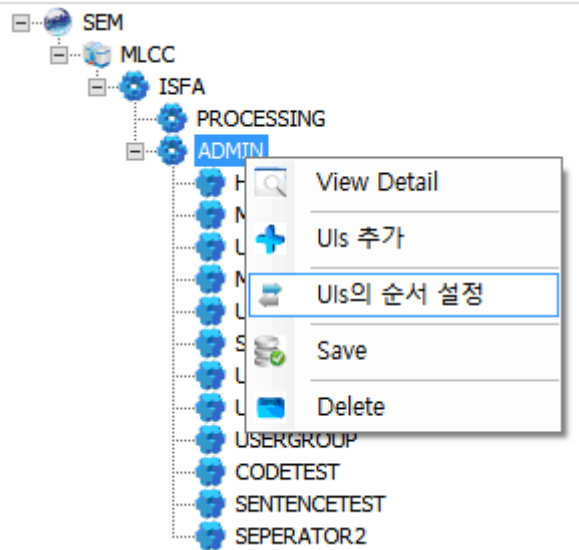
所有信息输入完成后, 点击 UI 左下侧的 [保存] 键. 所有 “SEPERATOR” 需安排在对应 Page Group 的最后面.

#### 3.5.2.4. 菜单顺序设定

菜单顺序在 [管理]-[UI阶层构造]上进行.

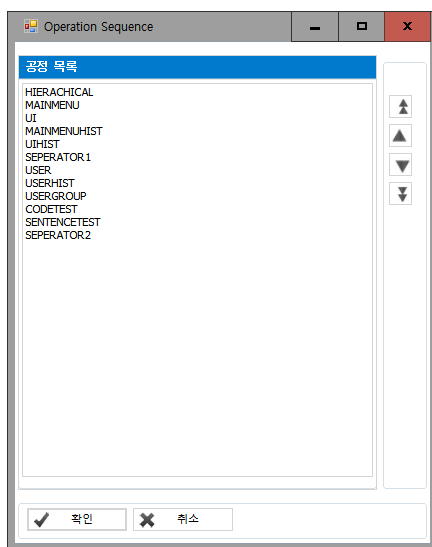
在树视图选择主菜单名称后, 选择点击鼠标右键所显示的 Context 菜单上的 [UI 的顺序设定].

([Fig3.5.2.4-1]).



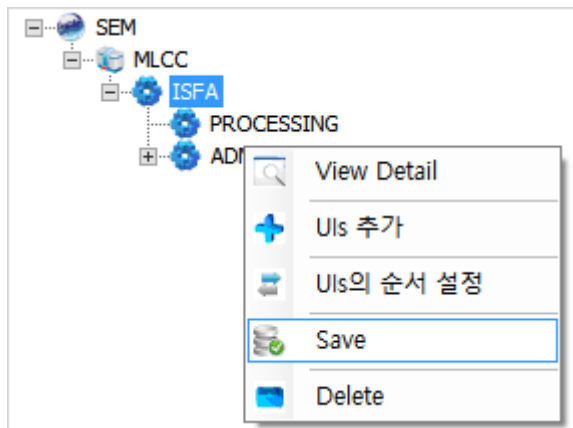
[Fig3.5.2.4-1]

跟[Fig3.5.1.3-2]一样使用设定界面右侧的箭头设定顺序.



[Fig3.5.2.4-2]

设定顺序完成且选择主菜单的名称后,点击鼠标键选择 Context 菜单上的([Fig3.5.2.4.-3])



[Fig3.5.2.4.-3]

## 4. Server-客户端通信

TAP FX都支持通过客户端与 DBMS直接通信的 2Tier方式和远程地址的 Host与 DBMS通信的 3Tier方式.在该 Section设定远程地址的 Host, 说明以 3Tier方式与 DBMS通信的方法.

### 4.1. Host 设置

作为 Server执行远程地址的电脑或者 Local电脑解除 tapfxat\_svr\_dev\_kit.zip文件的压缩.

打开 “TAP.Remoting.Server.Listener.exe.config” 文 Database Section.

```
<!-- 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. 使用开发用 Console Host

#### 4.2.1. Port设定

“TAP.Remoting.Server.ConsoleHost.exe.config” 文件的 Remote Listener Section跟 [Fig4.2.1-1]一样进行设定.

这时, 分配在 Listener Host Element与 Listener Element的 Port Key的端口号需解除防火墙. 如果, 根据报案政策上对该 Port不能解除防火墙, 对能够使用的 Ports的 Key值进行变更.

在当前版本上单个 Console Host可以对最多五个 Listener进行管理. 对 Listener Element的 EnabledKey的值分配 “true” 时, 对应的 Listener会被激活.



```
<!-- 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 修改

与 [Fig4.2.2-1]一样 “McWorksClient.exe.config” 文件 Environment Section的 InstallTypeKey值以 “CLIENT” 变更.

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

[Fig4.2.2-1]

#### 4.2.3. Remote Adapter Section 修改

与 [Fig4.2.2-1]一样 “McWorksClient.exe.config” 文件的 Remote Adapter Section的 HostIPKey值以 “localhost” 变更. Remote Adapter Section的 HostPortKey的值与 LocalPortKey的值分配的 Port需在防火墙上解除.

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

[Fig4.2.2-1]

如果, 根据测试环境的保安规定不能对对应的 Port解除防火墙,需变更 Port. 这时变更 Port时, “TAP.Remoting.Server.ConsoleHost.exe.config” 文件与 “TAP.Remoting.Server.Listener.exe.config” 文件的 Remote Listener Section内的端口号需变更成相同.

#### 4.2.4. 实行开发用 Console Host

实行 “TAP.Remoting.Server.ConsoleHost.exe” 文件.

```

*****
***** 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]

与 [Fig4.2.4-1]一样 Console Program实行时, 按 Enter键实行 Listener的话,另一个 Listener Console Program也会实行([Fig4.2.4-2], 在开发用 ConsoleHost上显示 Listener执行信息([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. 执行客户端

执行客户端执行文件(McWorksClient.exe)的话, Login窗口会实行,与 Listener Console的 [Fig4.2.5-1]一样显示实行Log.

```
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 TAPCTCODES
WHERE CATEGORY = 'FACILITY'
AND SUBCATEGORY = 'WX'
AND ISALIVE = 'YES'
```

[Fig4.2.5-1]

### 4.3. Server方呼叫 DB Method

Server方为了呼叫 DB Method需参照添加 TAP.Data.Client.dll. TAP.Data.Client.DataClient의 Instance使用提供的

Method呼叫 Server方 DB Method.

#### 4.3.1. 实行 Query

[code4.3.1.1-1]是开发人员实行 Query语句的例子. 生成 TAP.Data.Client.DataClient Instance后, 呼叫对应 Instance的 SelectData Method.结果值以 DataSet形式返还.

```
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. 实行 Non-Query

[code4.3.2-1]是开发人员实行 Non-Query语句的例子. 生成 TAP.Data.Client.DataClient의 Instance后, 呼叫对应 Instance的 ModifyData Method. 结果值以 int形式返还.

```
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. 实行 Transaction

[code4.3.2-1]是开发人员把 Non-Query以 Transaction实行的例子.生成 TAP.Data.Client.DataClient的 Instance를 后, Transaction要处理的 Non-Query语句积载到 String类型的 List上,呼叫 ModifyData Method.

```
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. 构成 Server Host

### 4.4.1. Server Host 设置

使用 SC.exe命令设置 Server Host Program.

4.4.1.1. 命令提示以管理人权限实行.

4.4.1.2. 在命令提示输入以下的命令语句.

```
sc create [Service名称] binPath= [Program Path]
```

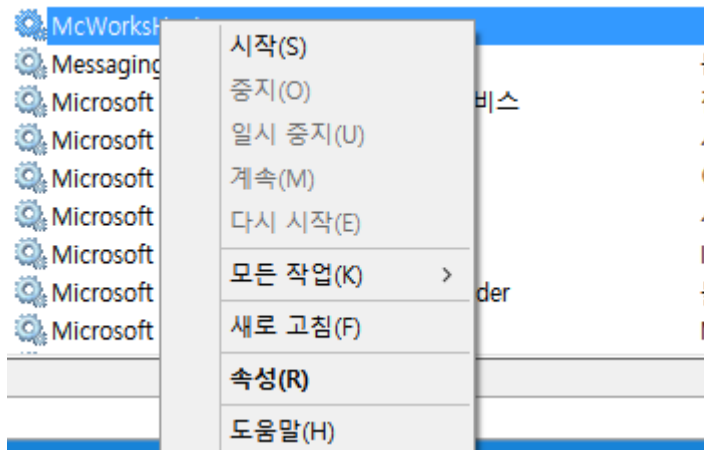
[Code4.4.1.2-1]

[Fig4.4.1.2-1]是在命令提示使用 sc.exe命令, 设置 Host的例子.

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

[Fig4.4.1.2-1]

4.4.1.3. 跟[Fig4.4.1.3-1]一样实行 Service 管理人员,开始 McWorksHost Service.



[Fig4.4.1.3-1]

#### 4.4.2. 客户端构成变更

与[Fig4.4.2-1]一样 “McWorksClient.exe.config” 文件的 Remote Adapter Section的 HostIPKey的值以 Host Program设置的服务器的 IP变更.

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

#### 4.4.3. 客户端执行及 Log 确认

实行客户端执行文件(McWorksClient.exe)的话, Login窗口会实行.

Server的动作状态是通过 Log文件确认. Log文件在 Host Program设置的 Directory的 “log\Remoting\remote\” Directory的下层 Directory储存([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. UI之间的通信

使用 TAP.UI.UIHelpBase.OpenAndCommand Method, 可以在特定的界面 Load 别的 UI, 或者可以呼叫其它界面的特定 Method. [Fig5.1]是 OpenAndCommand Method的 Signature.

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

[Fig5.1]

OpenAndCommand Method的各变量的意义与 [Table5.1]一样

变量	说明	备注
mainMenu	呼叫对象 UI的主要菜单名称	
menu	呼叫对象界面的菜单名称	
arguments	为了驱动呼叫对象界面的变量.	

[Table5.1]

该 Section为了 UI之间的通信,说明的技术方法.

### 5.1. 制作 OpenAndCommand Method

[Fig5.1.1]是使用 OpenAndCommand Method Load特定界面, 邀请对应界面的特定动作 ([Fig5.1.1]的 ArgumentPack参照本文档的 Section 6).

```
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. 制作 ExecuteCommand Method

TAP.UI.UIHelpBase.OpenAndCommand Method初始化呼叫对象界面后, 呼叫对象 UI的 ExecuteCommand Method. 从而 Load呼叫对象界面后, 在呼叫对象界面进行特定动作时, ExecuteCommand Method应该要存在.

[Fig5.2.1]是 ExecuteCommand Method制作的例子. 例子中的 ExecuteCommand Method以变量接收的 Lot Name在 Textbox上分配, 呼叫 UI内的 LoadDataAsync Method,Load Lot信息.

```
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. 传达复数的变量

在 TAP FX为了传达复数的变量,使用 ArgumentPack. ArgumentPack是复数的 Argument地 Collection.

### 6.1. Argument

Argument 个体与 [Table6.1.1]一样构成的.

构成要素	说明	备注
Name	在Argument Pack内,保障 Argument的唯一性的名称	
Type	Argument Value的类型	
Value	Argument의 Value	

[Table6.1]

### 6.2. ArgumentPack

与之前说明的一样, ArgumenPack是复数的 Argument的 Collection. Programing时生成 ArgumentPack的 Instance后, 呼叫AddArgument Method,鼓励在 ArgumentPack上添加 Argument Instance的方法.

[Fig6.2.1]是生成 ArgumentPack的 Instance, 添加叫 "LOTNAME" 的 Argument.

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

[Fig6.2.1]

### 6.3. 使用以变量接受的 ArgumentPack

因为 ArgumentPack是以 Key-Value形态构成, 使用 Key可以接近 ArgumentPack的特定 Argument.

[Fig6.3.1]是接近 ArgumentPack的特定 Argument, 带来 String 类型的值.

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

[Fig6.3.1]

Argument的值不是 String时,使用 ArgumentValue Property获取值,这时使用形变换代码. [Code6.3.1]是使用 ArguemntValue Property,获取值得几种例子.

```
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 Control 是为了在 UI 显示 Wafer里的各 DIE的特性而使用. Wafer Map Control 所支持的主要 Property 与 [Table 7.1.1.1] 一样.

Property	说明	备注
MapType	BINMAP, CD/THK Map 等的区分 Default: Value: BINMAP	
MapStyle	Square Map 或者 Circle Map 的区分 Wafer Map 时: Circle Map	
StartPointX	Map 左侧初始坐标值 Default Value: 11	
StratPointY	Map 上端初始坐标值: Default Value: 11	
MapData	获取 Map Data 值得 DataTable	
ShowDummyCell	设定 DIE 坐标 Data 以外的 Dummy Cell 显示与否	
ShowShot	设定 Shot 的显示与否	
ShowOuterLine	设定 Map 外围的显示与否	
ShowTitle	设定 Map Title 的显示与否	
Title	Map Title	
FlatZone	Map 常规方向 (T/L/R/B)	
MarginWafer	显示 Map 的基准点为止的余白值	

[Table7.1.1.1]

Property	说明	备注
<b>DieXColumn</b>	MapData 的 Die X 坐标值的字段名称	
<b>DieYColumn</b>	MapData 的 Die Y 坐标值的字段名称	
<b>PointXColumn</b>	MapData 的 Point X 坐标值的字段名称	
<b>PointYColumn</b>	MapData 的 Point Y 坐标值的字段名称	

[Table7.1.1.1]

Wafer Map Control 所支持的主要 Method 与 [Table7.1.2]一样.

Method	说明	备注
<b>CreateMap(DataTable <i>dt</i>)</b>	在 Map 绑定数据. dt: 在 Map Data 分配的 DataTable	
<b>SetShot( int <i>cellCntX</i>, int <i>cellCntY</i>, Color <i>color</i>, float <i>width</i>)</b>	设定 Shot. cellCntX: Shot X 坐标 cellCntY: Shot Y 坐标 color: Shot 边框颜色 width: Shot 边框颜色/粗细	

[Table7.1.1.2]

## 7.1.2. Data Scheme

[Table7.1.2.1]是 Map 常规信息的 Table Scheme.

Column	PK	Type	Description
--------	----	------	-------------

<b>FAB</b>	√	VARCHAR2(40)	
<b>TECH</b>	√	VARCHAR2(40)	
<b>LOT_CD</b>	√	VARCHAR2(40)	
<b>PROD</b>	√	VARCHAR2(40)	
<b>AREA_CD</b>	√	VARCHAR2(40)	
<b>OPER</b>	√	VARCHAR2(40)	
<b>DIE_X</b>		NUMBER	DIE 的横数量
<b>DIE_Y</b>		NUMBER	DIE 的竖数量
<b>MAP_X</b>		NUMBER	Map 的横大小
<b>MAP_Y</b>		NUMBER	Map 的竖大小
<b>FLAT_ZONE</b>		VARCHAR2(40)	
<b>SHOT_X</b>		NUMBER	Shot 的横 DIE 数量
<b>SHOT_Y</b>		NUMBER	Shot 的竖 DIE 数量
<b>UPDATE_DT_TM</b>		VARCHAR2(40)	
<b>MAP_TYPE</b>		VARCHAR2(40)	

[Table7.1.2.1]

[Table7.1.2.2]是 BIN Map 用 Map Data Table Scheme.

Column	PK	Type	Description
--------	----	------	-------------

<b>FAB</b>	√	VARCHAR2(40)	
<b>TECH</b>	√	VARCHAR2(40)	
<b>LOT_CD</b>	√	VARCHAR2(40)	
<b>PROD</b>	√	VARCHAR2(40)	
<b>AREA_CD</b>	√	VARCHAR2(40)	
<b>OPER</b>	√	VARCHAR2(40)	
<b>DIE_X</b>		NUMBER	DIE的 X 坐标
<b>DIE_Y</b>		NUMBER	DIE的 Y 坐标
<b>MAP_VAL</b>		VARCHAR2(40)	
<b>UPDATE_DT_TM</b>		VARCHAR2(40)	

[Table7.1.2.1]

[Table7.1.2.2]是 THK, CD Map Data Table Scheme.

<b>Column</b>	<b>PK</b>	<b>Type</b>	<b>Description</b>
<b>FAB</b>	√	VARCHAR2(40)	
<b>TECH</b>	√	VARCHAR2(40)	
<b>LOT_CD</b>	√	VARCHAR2(40)	
<b>PROD</b>	√	VARCHAR2(40)	
<b>AREA_CD</b>	√	VARCHAR2(40)	
<b>OPER</b>	√	VARCHAR2(40)	
<b>DIE_X</b>		NUMBER	DIE的 X 坐标
<b>DIE_Y</b>		NUMBER	DIE的 Y 坐标
<b>POINT_X</b>		NUMBER	DIE里的 X 坐标位置
<b>POINT_Y</b>		NUMBER	DIE里的 Y 坐标位置
<b>MAP_VAL</b>		VARCHAR2(40)	
<b>UPDATE_DT_TM</b>		VARCHAR2(40)	

[Table7.1.2.2]

## 8. Namespace Naming Rules

### 8.1. 概要

Namespace遵循以下命名规则.

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

各单元功能 (Service 或 UI)在上层定义的 Namespace 后面进行命名.在每个开发环境的公用/共同/上层 Namespace 时,可省略 “Main Menu (Service Group)” .

### 8.2. Prefix

Prefixs 选择 [Table8.2-1] 里列出的 Prefix 中的一个进行匹配使用.

Prefix	Description	Remark
TAP	TAP Framework Namespace	TAP.Base.Database
IntegratedSystems.	Package 产品用 Namespace	IntegratedSystems.ISEM.Equipment
ISTS	Technical Service用 Namespace	ISTS.HWQQSEIS.Common

[Table8.2.-2]

### 8.3. System Name

#### 8.3.1. Package 产品

Package 产品时, 使用对应产品的名称(例: ISEM, ISM).

#### 8.3.2. Technical Service

如果是 Technical Service 时, 使用客户端代码 + 系统组合(例: HWQQNEIS). 但针对 Technical Service 时,可以使用客户需求的名称.

## 8.4. AssemblyInfo

AssemblyInfo 的每个项目的制作跟 [Table8.4-1]一样.

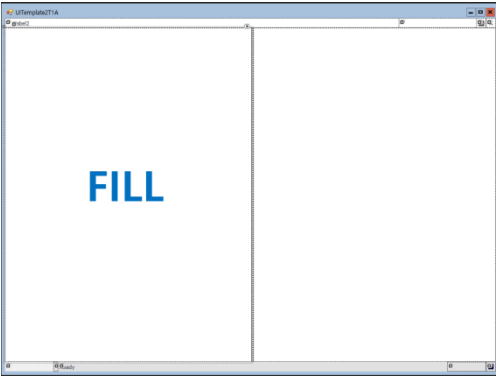
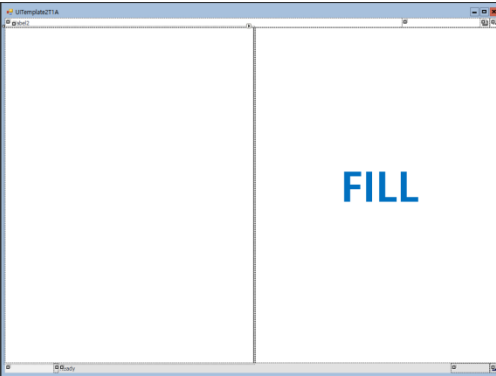
Item	Description	Remark
<b>Assembly Title</b>	对应 DLL 的名称或者 Main Namespace	TAP.Base.Database
<b>Assembly Description</b>	-	
<b>Assembly Configuration</b>	-	
<b>Assembly Company</b>	ISSET-DA Shanghai Co., Ltd.	
<b>Assembly Product</b>	Package: Package 名称.	ISM, ISEM
	Technical Service: 时显示 Project 名称	
<b>Assembly Copyright</b>	Package: ©iSET-DA Shanghai 2011-2022	开发开始时间年度~当前年度
	Technical Service: 根据项目环境	
<b>Assembly Trademark</b>	-	
<b>Assembly Information Version</b>	以 x.x.x.x 的形式, 管理方便而制作	

[Table8.2.-2]

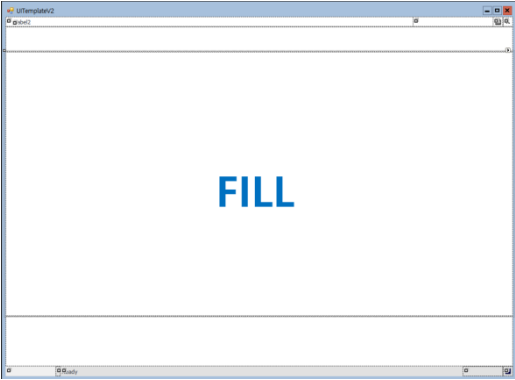
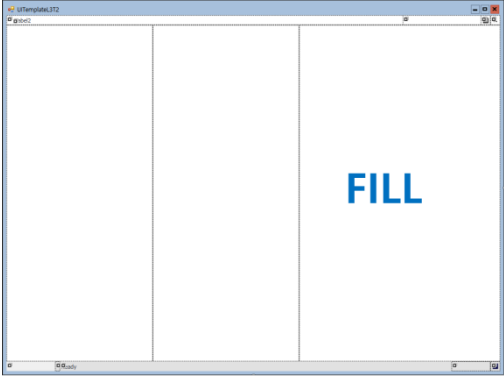
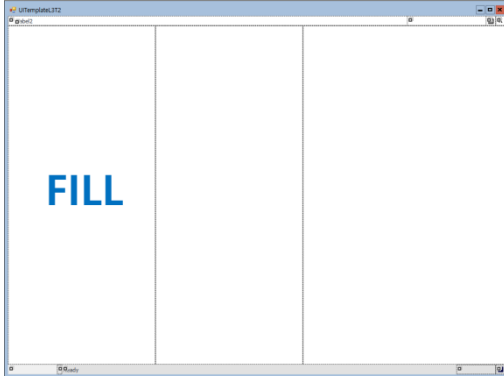


# Appendix 1. UI Templates

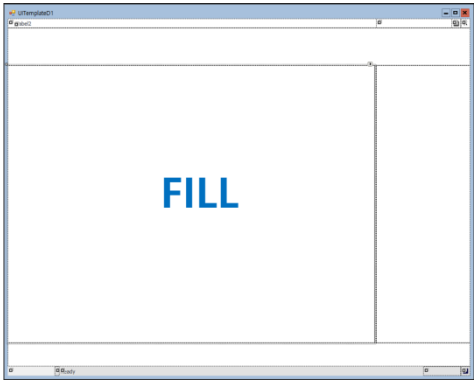
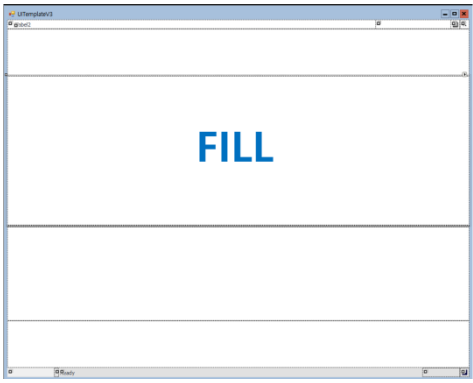
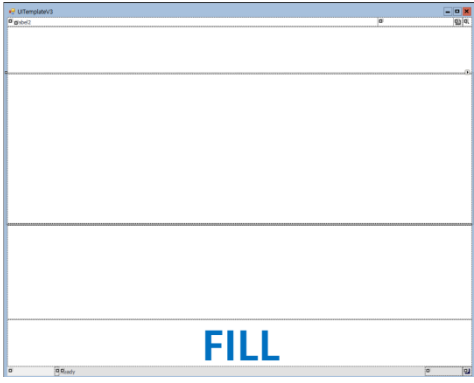
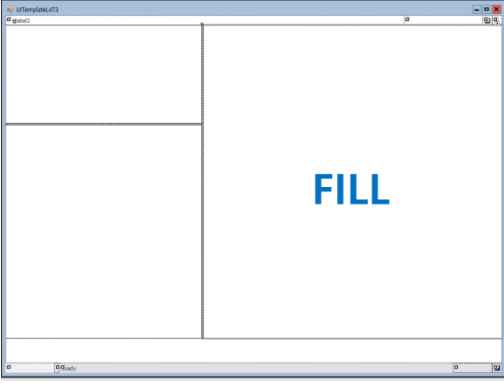
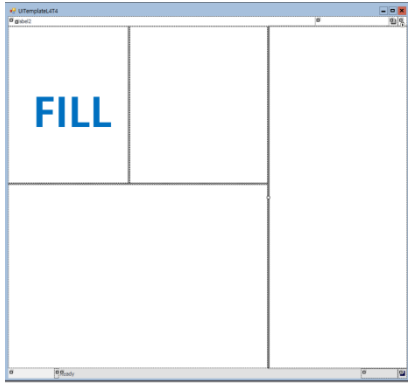
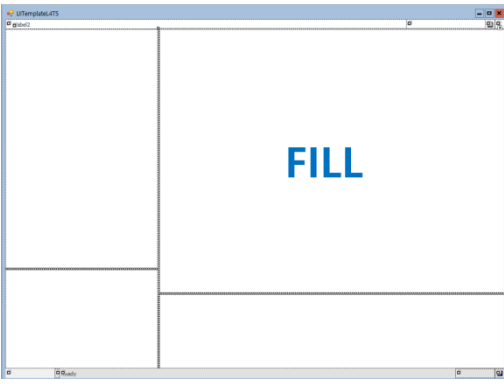
## Appendix 1.1 2Layer Templates

UI Template 2T1A	UI Template 2T1B
	

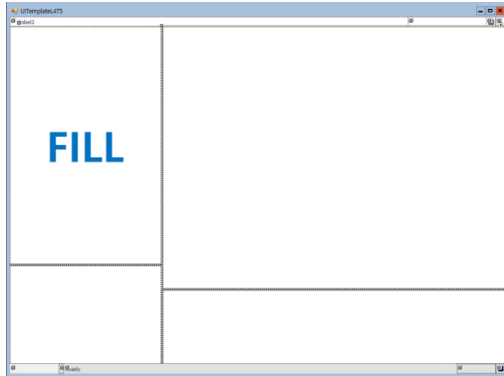
## Appendix 1.2 3Layer Templates

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

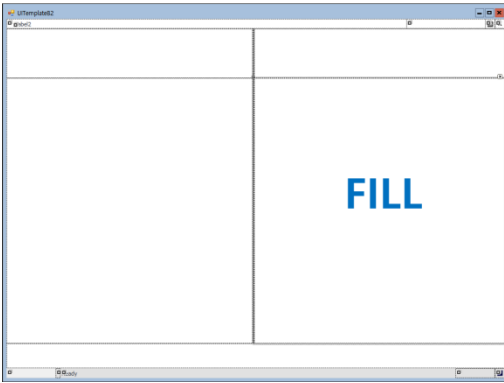
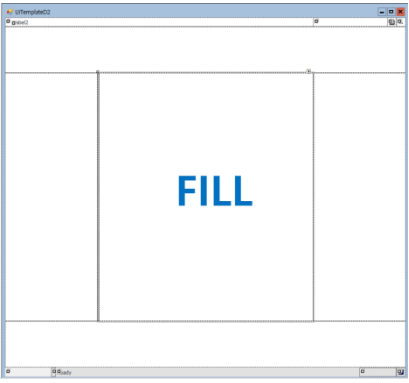
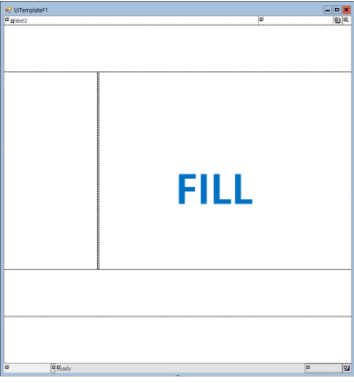
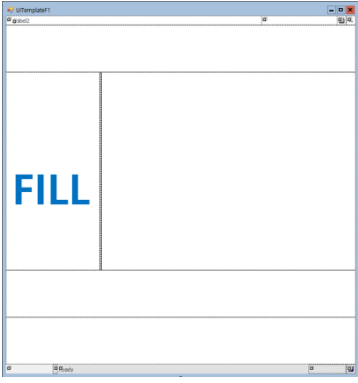
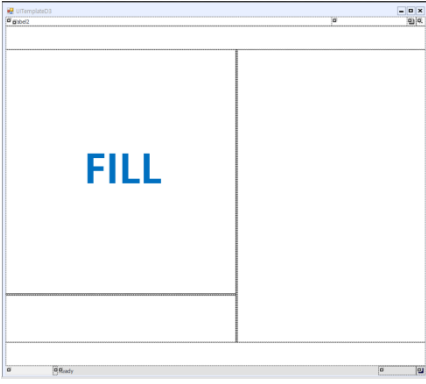
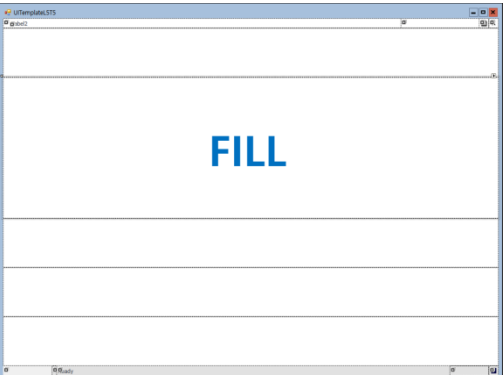
## Appendix 1.3 4Layer Templates

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

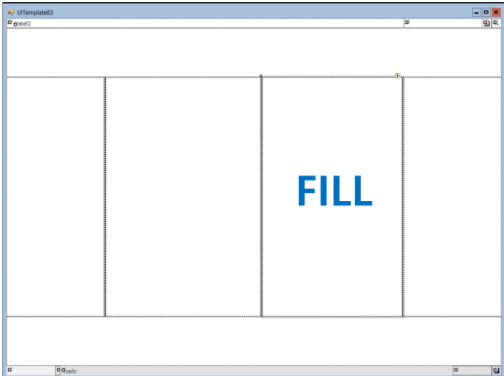
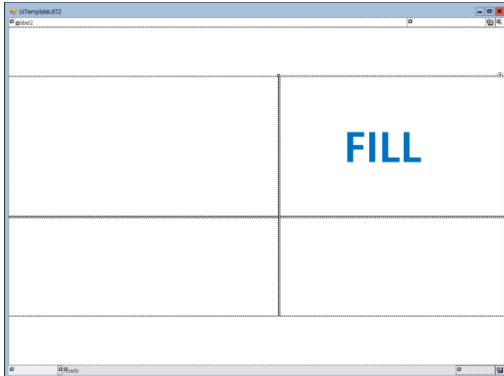
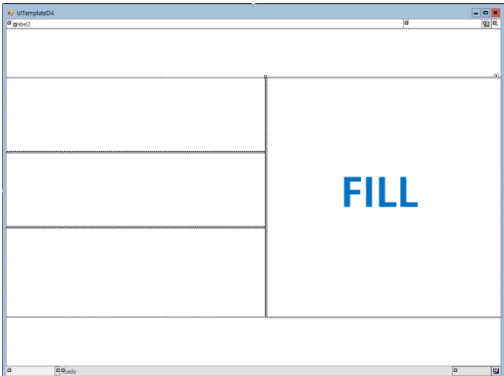
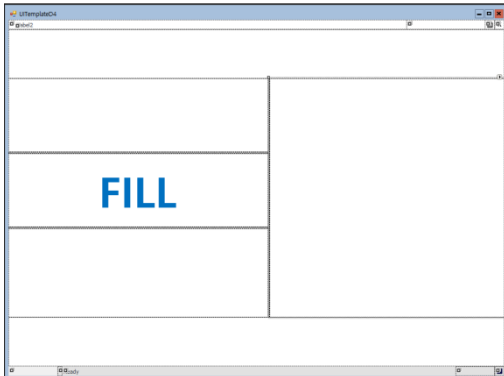
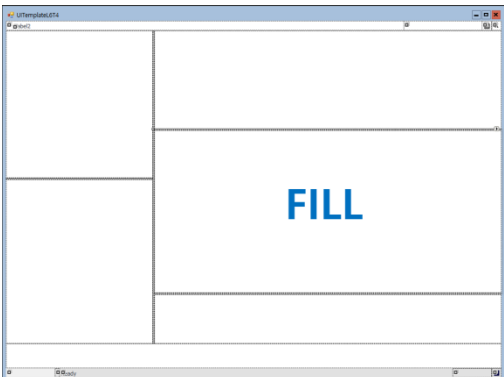
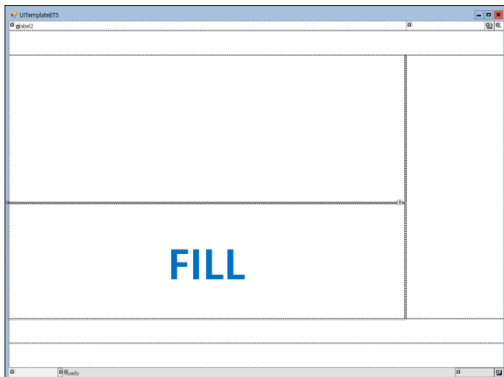
## UI Template 4T5A



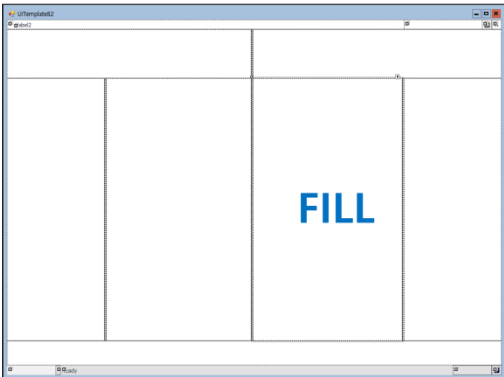
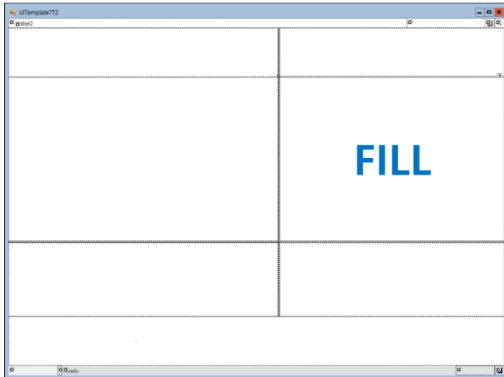
## Appendix 1.4 5Layer Templates

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. 标准代码制作方法

### Appendix 2.1. 共同事项

制作 Source Code时,需遵守以下事项.

- 除了常量以外的 Namespace, Class及所有成员的名称,使用 Camel Case.
- 所有 Public Member需添加 XML形式的的注释, 用英语作成简单的说明.

### Appendix 2.2 成员种类别配置顺序

Importing目录的情况, 与 [FigA2.2-1]一样在上端列出 .NET Framework 参照的 Namespace,在下面列出 TAP FX 参照 Namespace.

在 Class内部与 [FigA2.2-1]一样,以 Member种类别使用 #region指示者捆绑. 成员种类别列出顺序如下.

- 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 及 Property声明

### Appendix 2.3.1. Constant 声明

声明 Constant时,除了特殊情况以外,使用 Public限定字. Constant的 Naming Rule根据以下内容.

- 基本都是用大写.
- 需要时名称中间使用 “\_” .
- Constant的名称以 “\_” 开始.

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

[Fig A2.3.1-1]

### Appendix 2.3.2. Field 声明

声明 Field时除了特殊情况以外,使用 private限定字. Field의 Naming Rule根据以下内容.

- Field的名称以 “\_” 开始.
- Field名称的第一个字使用小写.

```
#region Fields

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

private new string _excelFilePath;

#endregion
```

[Fig A2.2-1]

### Appendix 2.3.3. Property 声明

声明 Property时,除了特别的情况以外需要使用 public限定字. Property的 Naming Rule根据如下.

- Property 名称的第一个字使用 “大写” .

```
/// <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]

而且所有 public及 protected成员上添加注释, 注释的内容以英文作成.

## Appendix 2.4. Creator와 Disposer 制作

### Appendix 2.4.1. Creator 制作

Creator的内容与 [Fig A2.4.1-1]一样,通过 #region处理方式捆绑. 而且, Creator以 “InitializeComponent” 进行呼叫以外的动作时,与 [Fig A2.4.1-2]一样,使用 try-catch句.

```
/// <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]

使用 Creator内的局部变量时, 局部变量在代码的开始部分(try-catch句外部)声明.

## Appendix 2.4.2 Disposer 制作

由于.NET Framework的特性 Disposer不必要制作, 但制作 Disposer情况时,是根据以下 RULE.

- Disposer的内容使用 #region Indicator 捆绑([Fig A2.4.2-1]).
- Disposer的内容使用 try-catch句.
- 在 catch句避免发生个别的 Exception(需要时要留下 Log).

```

/// <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 Method及 Method制作

制作 Method时,遵守以下事项. Method的制作例子参考 [Fig A2.5-1].

- Method的名称必须以大写开始.
- 全部代码的内容以 #region处理方式捆包.
- 全部逻辑代码以 try-catch句捆绑.
- 局部变量在 Method的开始部分声明, 在try-catch句型外声明.
- 初始化 Method时使用 Initialize名称.

```

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 制作

制作 Event Handler时,遵守以下事项. Event Handler制作例子参照[Fig A2.6-1].

- 在 Event Handler以呼叫执行实际动作的 Method程度进行简单制作.
- 全部逻辑以 try-catch句捆绑.
- 在 catch句制作具体的例外处理代码(Popup, Logging等)

```

1 reference
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

Control的名称是以意味 Control的种类的 Prefix和 Control的作用的名称组合作成.

比如输入用户名称的 Textbox Control的话, 组合意味 Textbox的 Prefix "txt" 与 意味对应Textbox作用的 "UserName" 跟 "txtUserName" 一样的形式作成.

显示 Control种类的 Prefix都以小写使用,显示 Control作用的名称以大写开始的 Camel Case.

[Table A2.7-1]是主要 Control的 Prefix.

.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. 设置问题点及解决方法

### Appendix 3.1. 登录窗口不显示 Facility 信息时或者 Log In 窗口显示同时发生 System.Null.Exception 时

Database 连接发生错误时,会发生对应的现象.

Database 连接发生错误时,解压添加的文件 “[odp.net4.zip](#)”, 解压的文件夹内的 (bin, odp.net)文件夹复制粘贴到 Oracle 安装文件夹里边.

### Appendix 3.2. 设置服务后服务无法正常运作时

Database 连接字符串不正确时,会发生对应的问题.

把正确的数据库字符串输入到以下 Configuration 文件的 DB Section上.

- TAP.Remoting.Server.ConsoleHost.exe.config
- TAP.Remoting.Server.NTHost.exe.config
- TAP.Remoting.Server.Listener.exe.config

### Appendix 3.3. 多语言 Loading 或者连接 Access DB 发生错误时

出现该问题时设置已添加的文件 “[AccessDatabaseEngine\\_X64.exe](#)” 后, 重新运行程序.

### Appendix 3.4. 使用OracleManagedDataAccess时无法连接数据库的情况

连接OracleManagedDataAccess时, 在FX Configuration文件Database Section/Connection Element的ConnectionString Key输入 “所有连接信息”, 而不是TNS Alias ([FigA3.4-1].

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

[Fig3.4-1]