# ConfMgmt(Xml载体)快速指南

## 架构简介

ConfMgmt解析指定目录下所有xml文件，将所有xml节点集中管理。

ConfMgmt总共分三层：

* ConfMgmt对应目录（可以包含多级子目录），
* ConfTree对应xml文件，
* ConfItem对应配置项。

## 读配置项

### 默认ConfMgmt生成

*ConfMgmt.Default.Generate(@"Configs\Configuration");*

### 指定ConfMgmt生成(适用于两个及以上的目录)

*ConfMgmt.Inst("ProductFile").Generate(@"Configs\ProductFiles");*

### 获取ConfItem

GetItem对ConfMgmt下所有元素进行遍历查找，相对低效。

获取ConfItem需要指定Item路径，包含以下三种格式：

#### 基本获取（适用于具有唯一item名称的ConfItem）

直接使用ConfItem的名称

*ConfMgmt.Default.GetItem(@"channel")*

#### 指定路径获取（适用于item名称非唯一的ConfItem）

使用/逐级分割

*ConfMgmt.Default.GetItem(@"Trigger/channel")*

#### 指定Tag获取（适用于查找具有相同ConfTree结构中的ConfItem）

使用:指定tag

*ConfMgmt.Default.GetItem($"HighTemp:TempDev")*

### 获取ConfTree

var *tree = ConfMgmt.Default.GetTree("Hardware")*

#### 基于ConfTree获取ConfItem

var *itemValue = tree[item路径]*

## 改配置项

### 基于ConfTree修改ConfItem

1. [获取ConfTree](#_获取ConfTree)
2. *tree[item路径] = newValue*
3. *tree.Save()*

### 基于ConfTree创建新的ConfTree

*var newconf = ConfMgmt.Inst("ProductFile").Root[SelectedPath].Clone() as ConfTree;*

*newconf.Save(path);*

# ConfViews

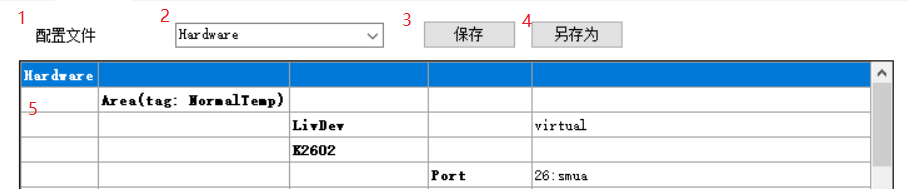
### ConfSelector(ComboBox)



void Bind(ConfMgmt conf)

提供指定目录（ConfMgmt）下所有配置文件选项

### ConfMgmtView



void Bind(string title, string confType, string path, Action onUpdate = null)

ConfMgmtView用于管理配置文件。使用Bind函数使之与某个配置目录相关联：

* title：指定1的显示内容
* confType：指定ConfMgmt的Inst名称（系统唯一）
* path：指定配置文件所在目录
* onUpdate：在配置项发生变化时需要执行的回调函数

### ConfUC

ConfUC（继承自UserControl）支持使用者自行绘制与ConfTree相关联的GUI界面。ConfUC使用以下两个接口，与ConfTree交互配置项信息。其中，Apply用于将ConfTree的各个配置项展示到GUI，Generate根据GUI界面中的设置值生成ConfTree。

public void Apply(ConfTree conf)

public ConfTree Generate(string tag = null)

#### ConfUC命名规则

TextBox的名称与相关联的配置项名称保持一致。

RadioButton的名称为相关联的配置项名称后加数字（配置项需指定guitype="RadioButton"）。

# 窗口编辑器打开失败的问题



## 原因

引用工程的Utils版本与ConfMgmt的Utils版本不一致

## 解决方法

将ConfMgmt引用的Utils改为引用方案的Utils。

