

AgileEAS.NET SOA 平台 界面设计器使用教程



敏捷软件工程实例室

2013 年 01 月 26 日

文件状态： [] 草稿 [√] 正式发布 [] 正在修改	文件名称：	AgileEAS.NET SOA 平台界面设计器使用教程
	当前版本：	3.0
	作 者：	魏琼东
	完成日期：	2011 年 01 月 26 日

目 录

前言.....	3
关于界面模型.....	4
系统主界面.....	5
系统菜单.....	5
工具栏.....	5
导航栏.....	6
工作区.....	6
状态栏.....	6
关于.....	6
界面模型解决方案.....	7
概述.....	7
新建项目.....	7
保存项目.....	8
打开项目.....	9
界面模型定义.....	11
概述.....	11
界面模型组织.....	11
新建界面模型.....	11
删除界面模型.....	12
定义界面模型.....	12
设置页面布局.....	18
打开界面模型.....	22
关于界面布局.....	23
界面解决方案输出.....	24
概述.....	24
输出代码.....	24
WinForm 界面代码输出.....	25
Web 解决方案输出.....	30
WPF 解决方案输出.....	35
结束语.....	41
联系我们.....	41

前言

AgileEAS.NET 平台做为一个快速应用开发平台，其目的为是为了提高应用软件的生产效率，如何软件开发的效率，方法是多种多样的；使用工作简化开发中低技术重复工作可以是一种行之有效的途径。

在 AgileEAS.NET 平台中，我们提供了一个集界面设计、代码生成、解决方案定义于一体的界面设计器。

在 AgileEAS.NET5.0 的最新版本中新增了该功能，界面设计器的主旨是通过一次设计，然后生成不同界面技术上的呈现方案，例如现在技术上主要是 C/S,B/S，设计器提供了对这二种方式的支持，目前支持的代码生成的解决方案支持：Winfrom、Webfrom、WPF、SL 四种。

在基于数据库的应用开发项目中，数据库设计是很一个很重要的过程，而这个过程写数据库设计文件是一个环节，在很多软件公司中，都是使用 Word、WPS 等文字表格工具写数据库文档，但是数据库文档与数据库建立、程序编写过程脱节。

AgileEAS.NET 平台的对象设计工作，试图在这方面进行集成，即设计器产生的模型定义即可以生成数据库文档、数据库定义语句、也可以生成开发过程的代码。

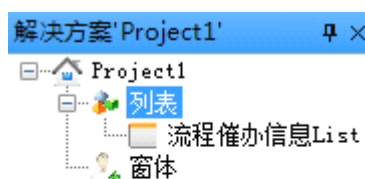
AgileEAS.NET 平台的界面设计器，在前面的对象设计器的基础上产生的解决方案，可以被界面设计器打开，实现了，一次解决方案的建立，多种工具的通用和集成工作，这样更能方便和提供开发过程，并且能够生成设置后的解决方案代码。

关于界面模型

AgileEAS.NET 平台针对应用开发之中的数据对象定义、存储和交流定义一个数据对象结构模型定义文件，数据设计器建立好的项目定义最后存储在以.sdm 为扩展名一个数据模型定义文件。

AgileEAS.NET 平台的界面设计器则是在数据模型文件的基础上打开或者新建自己的解决方案，后缀名也是.sdm 文件。

AgileEAS.NET 平台的界面模型定义如下：



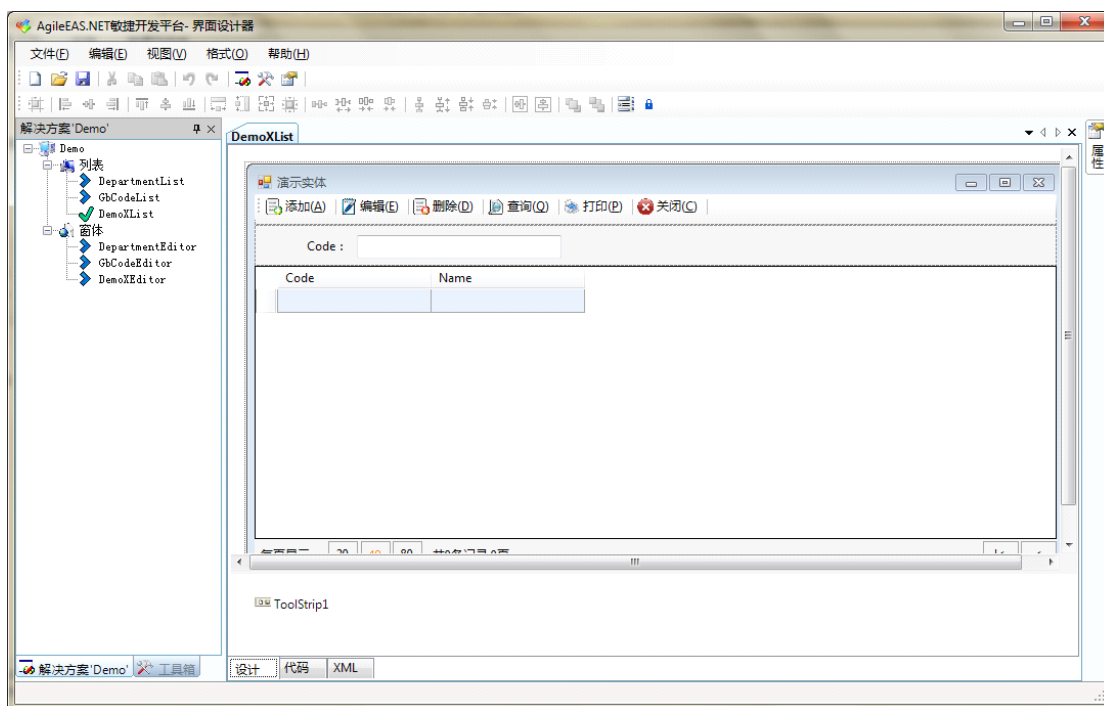
其组织结构为项目包含列表和窗体两大类，列表主要是查询功能的相关界面，窗体的主要是编辑、新增页面。

系统主界面

首先、找到平台的开发包内部的如下应用程序。



双击打开后，我们看到设计器的主界面。



主窗口中共包括了 5 个不同的工作区：系统菜单、工具栏、功能导航栏、业务工作区、系统状态栏，系统中的所有业务功能均可通过系功能导航栏访问操作。

系统菜单

界面设计器采用导航式界面样式，系统功能由导航和菜单两部分组成，提供文件、编辑、视图、调试、窗口和帮助等菜单。

工具栏

工作栏上放置了与系统菜单相关的快速工具栏，可以通过这些工具栏快速访问相关功能模块。

导航栏

系统导航栏上列举了项目中的界面设计模型的列表，主要是列表和窗体。

导航栏默认显示在界面的左边，如果你想让业务工作区更大些，以便有更大的界面空间处理业务功能，你可以在进入相关的业务功能模块后，通过系统菜单或工具栏的导航命令隐藏或显示功能导航栏。

工作区

工作区是系统工作区域，工作区根据导航和菜单的不同选择与操作，将会加载不同的功能模块，用于完成必要的任务。

状态栏

系统状态栏上显示了系统当前的处理任务及任务处理状态，用于通知用户，以便及时了解系统的运行情况。

关于



提供对当前应用程序的描述信息。包括产品名称、版本、开发商、授权信息等。

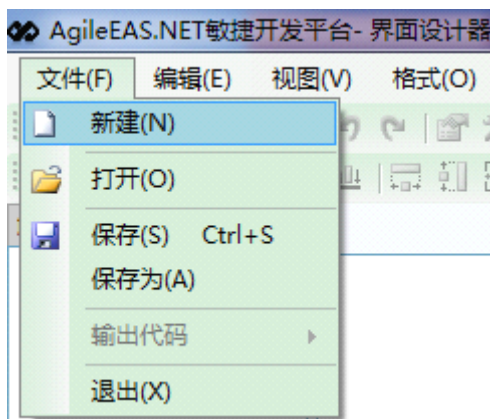
界面模型解决方案

概述

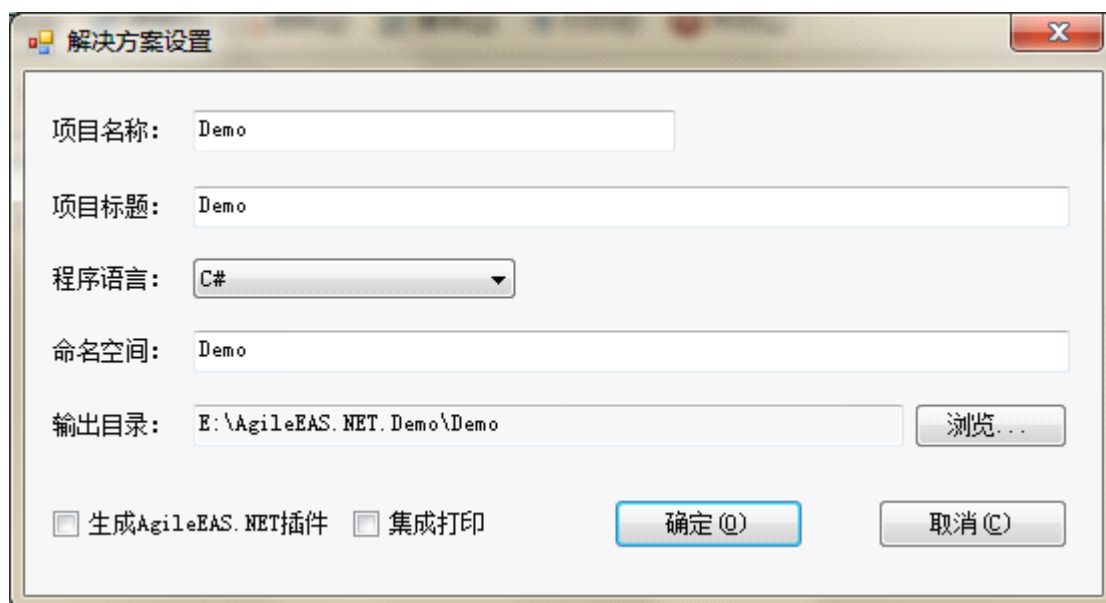
在进行设计界面模型之前，需要先规划自己的项目以及项目中的项目组织，预定义好的数据对象项目解决保存在硬盘上，以方便开发人员的交流和以后的修改。并且界面设计器在基于之前定义好的数据对象解决方案的基础之上来构建界面模型解决方案

新建项目

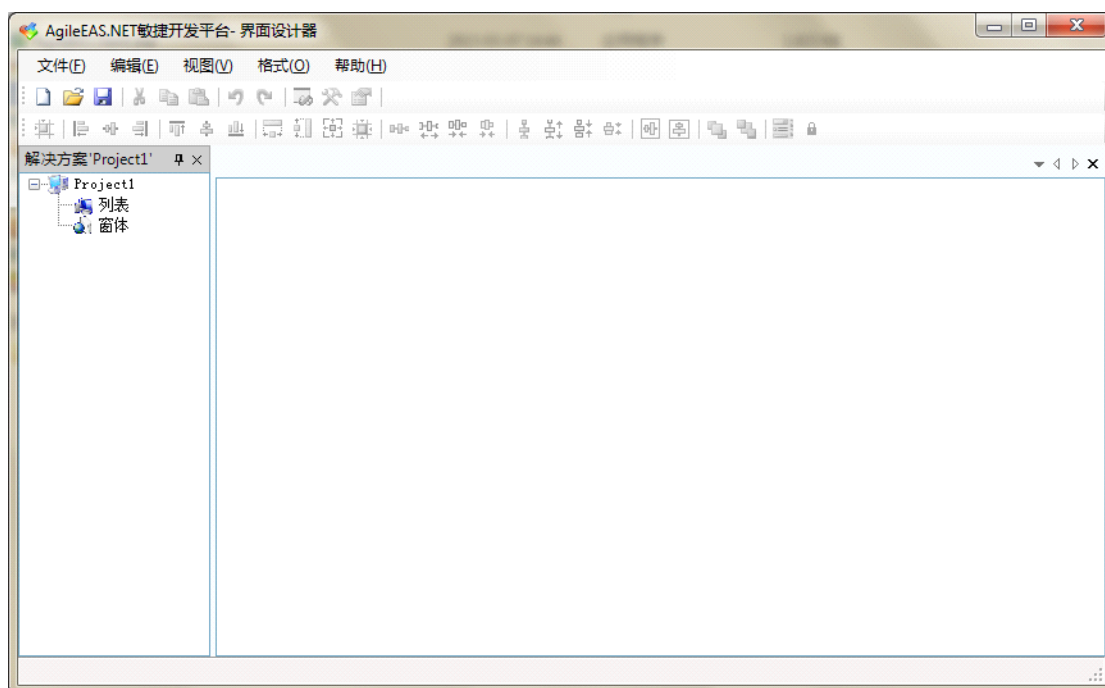
打开文件菜单的“新建”按钮。



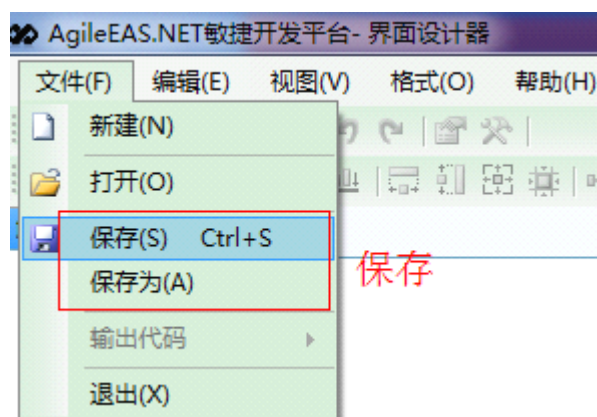
然后出现如下界面：输入解决方案的名称、命名空间和解决方案的存放位置，解决方案的说明信息（非必填）等。



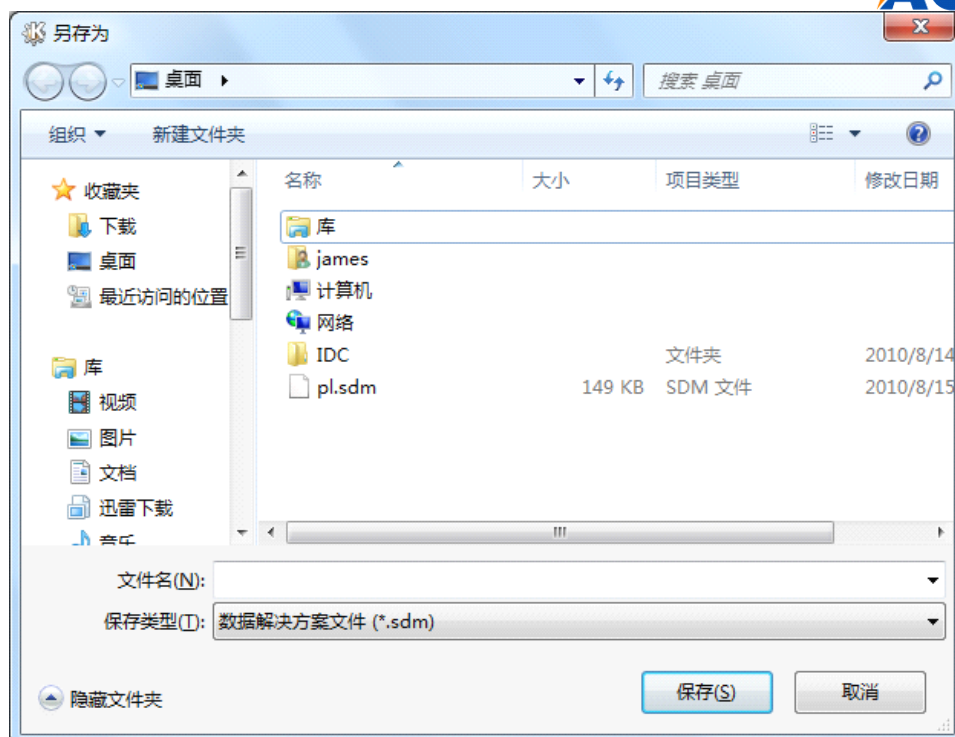
输入信息后，点击“确认”按钮，系统会默认重置页面的布局，效果如下：



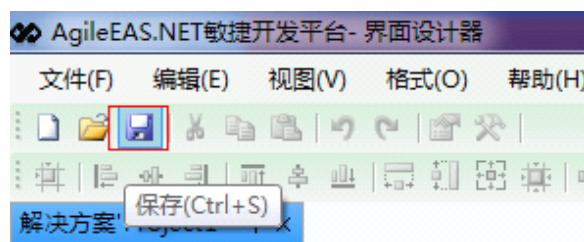
保存项目



使用文件菜单中的保存解决方案按钮或另存按钮，则会保存界面模型解决方案的信息，保存的路径即是新建解决方案时的存放路径或是打开解决方案时的路径。

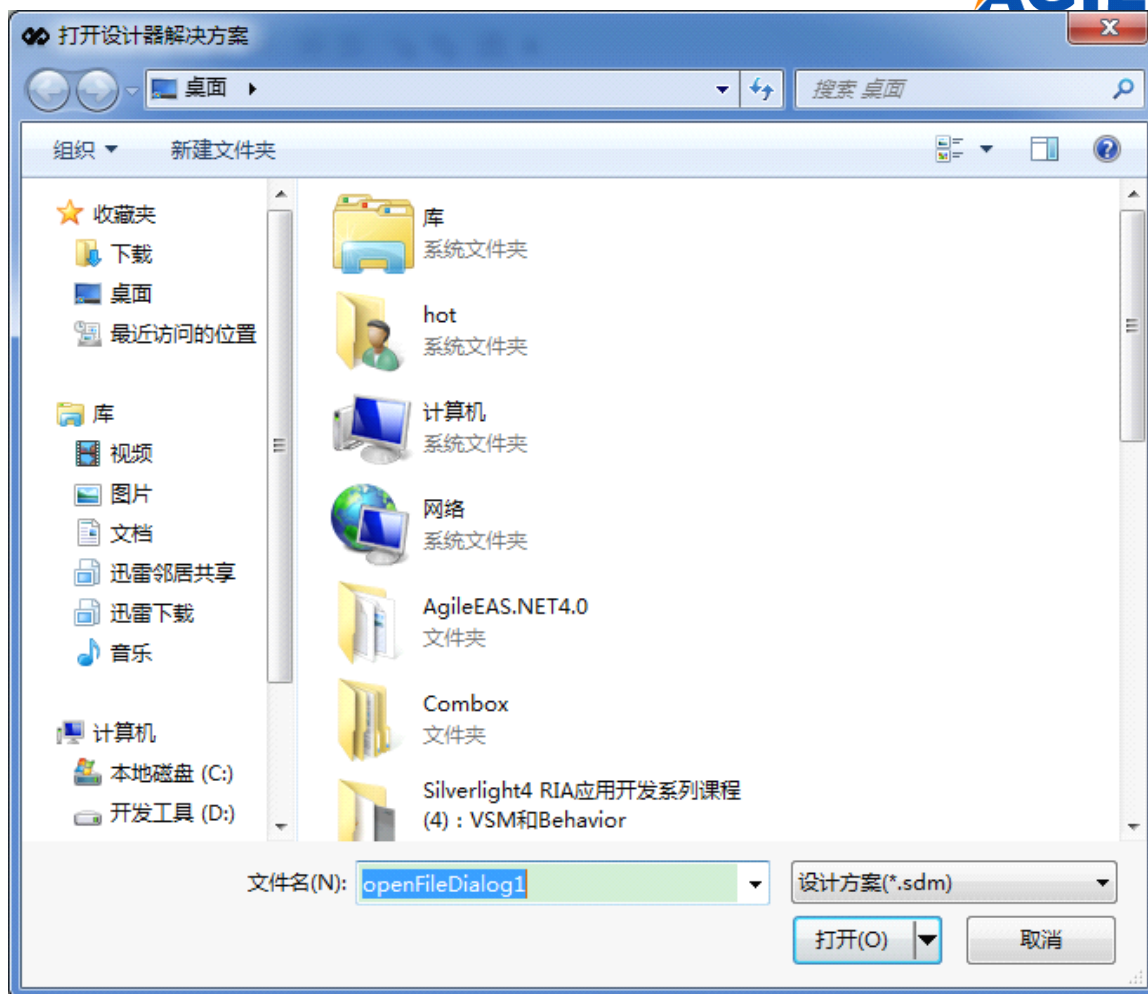


还有一直保存方式是通过快捷菜单中的保存按钮来保存。

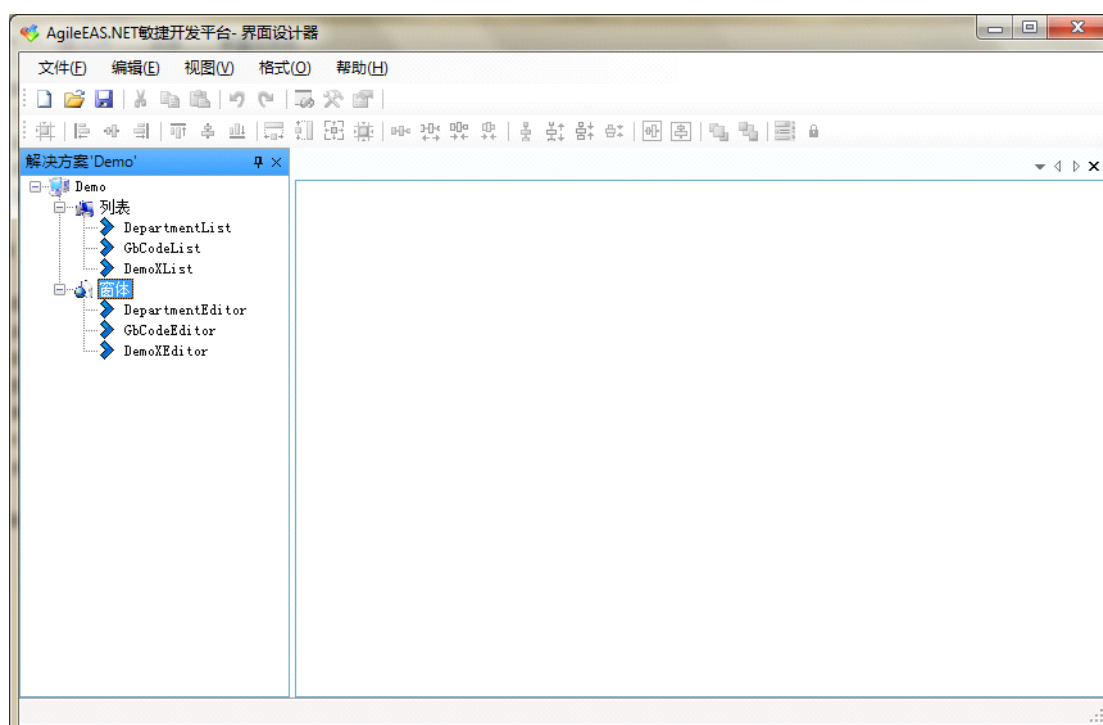


打开项目

使用文件菜单中的“打开解决方案”按钮，弹出文件打开对话框：



选择文件系统已经存在的界面模型项目文件并打开，系统根据模型文件重置导航栏和清空工作区：



界面模型定义

概述

定义界面模型是界面设计器中最重要的工作，在模型定义信息中，存储了界面的布局信息与表单的基本设置信息，包括具体的每个实体属性与控件的映射信息及界面控件的类型等等基础的界面信息。

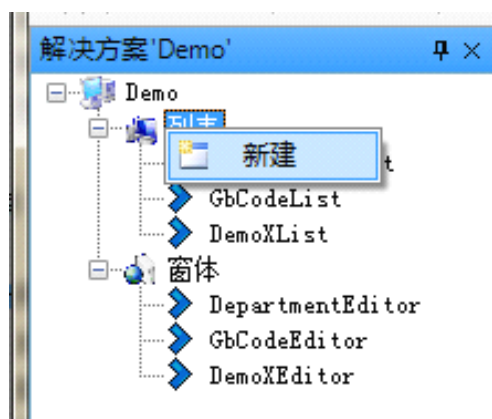
界面设计器设计成的模型用与生成项目的界面设计文件与界面后台代码文件。

界面模型组织

界面设计器中提供了一个列表与窗体的概念，即对于界面对象进行分组管理，系统会默认根据新增的是窗体还是列表默认加入指定的分组。

新建界面模型

在导航栏中的列表分组或窗体分组上“右键鼠标”新建界面设计来添加界面模型：



列表分组上点击“鼠标右键”只能添加列表。

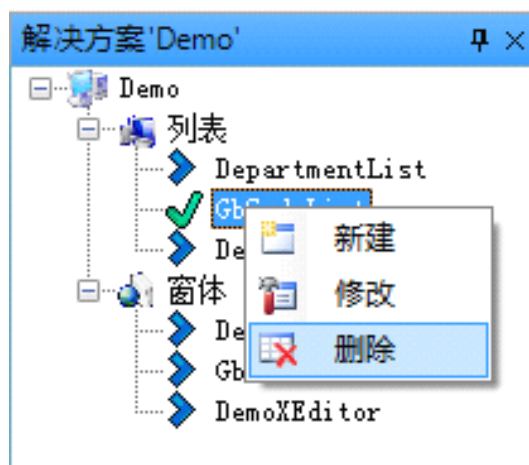
窗体分组上点击“鼠标右键”只能添加窗体。

目前系统中支持的二类模型，一种是窗体，一种是列表，基本上可以满足目前 80%以上的界面设计需求。

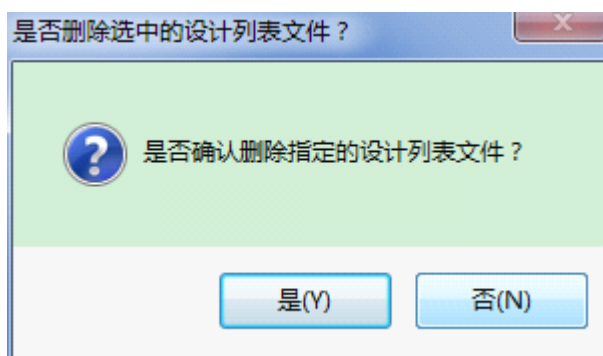
后续会考虑增加符合界面应用。

删除界面模型

选中要删除的界面模型，点击“鼠标右键”：



弹出的菜单中选择“删除”，系统会提示您是否确认删除，点击“是”删除相应的界面模型。

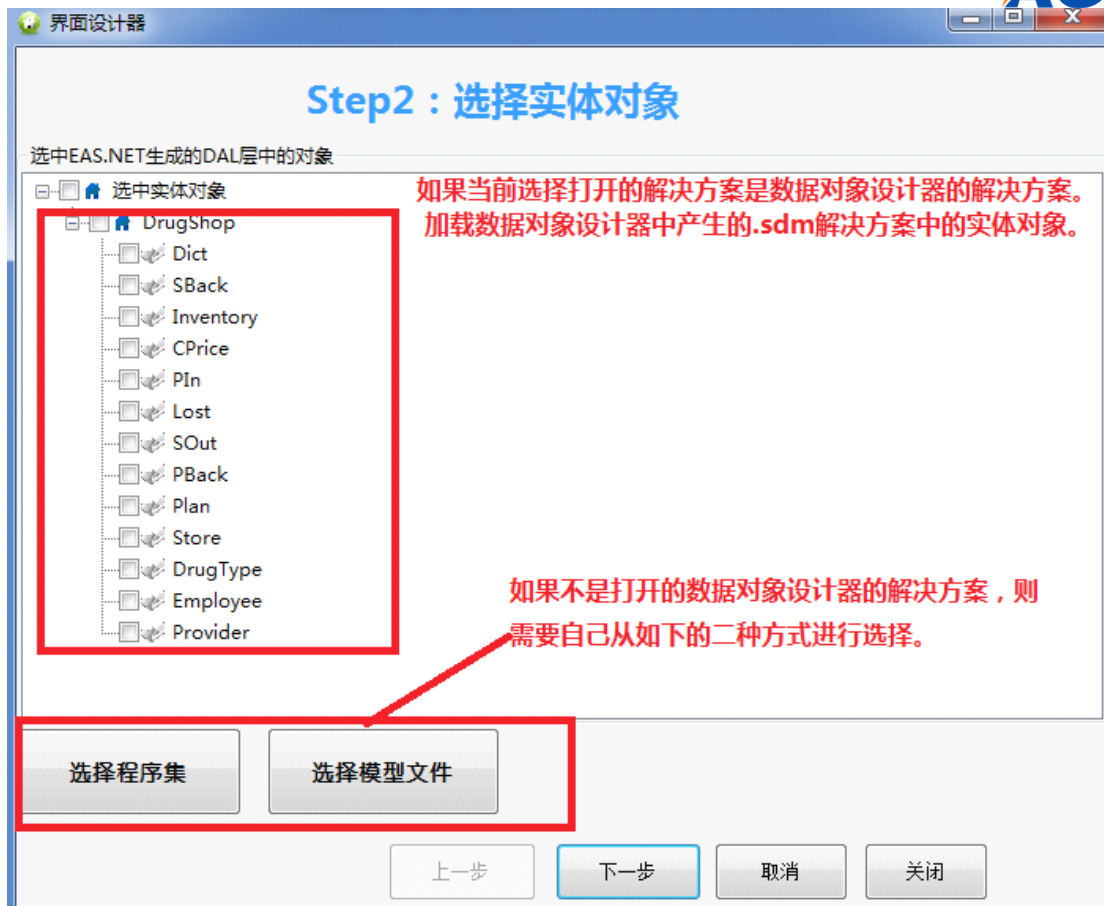


注意：删除界面模型后，将无法恢复。

定义界面模型

操作参考“新建界面模型”小节。

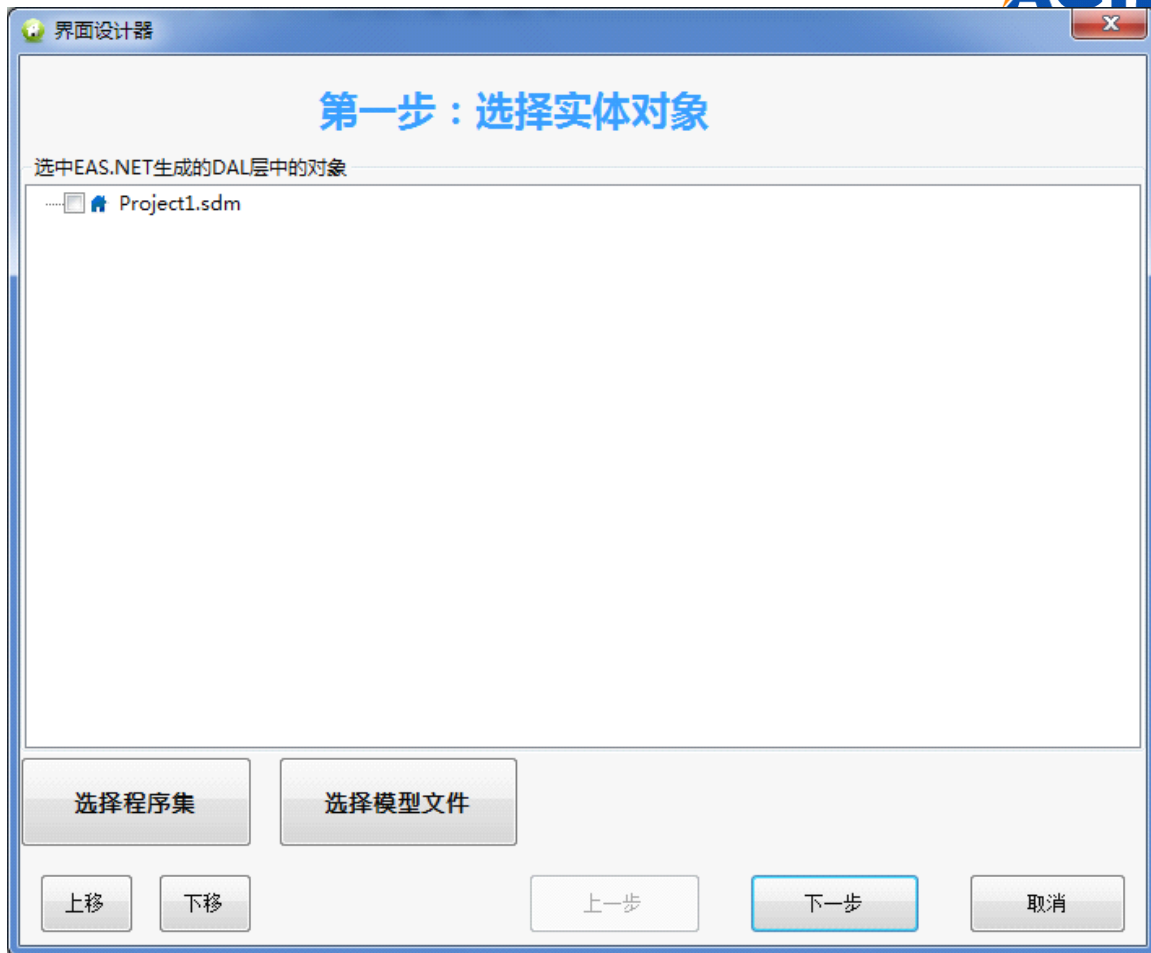
点击“新建列表”，弹出如下对话框：



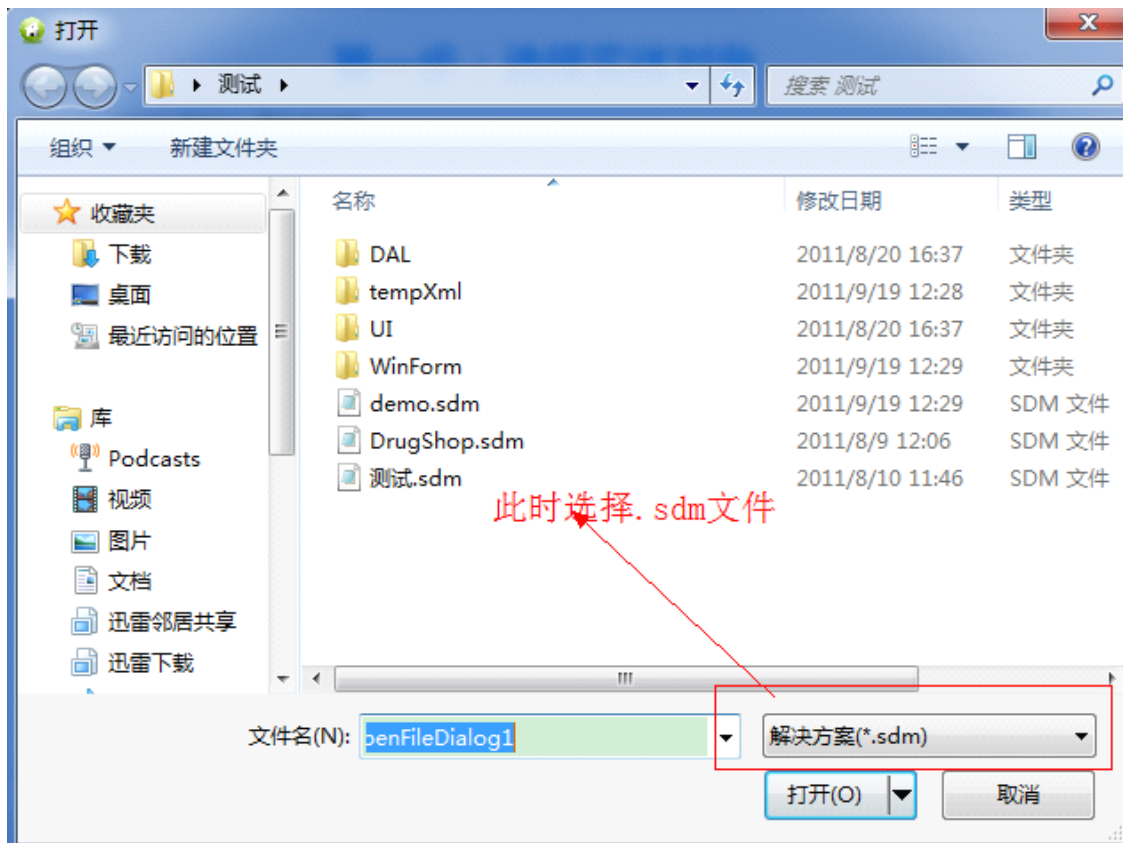
请注意：当我们的界面设计器打开的方案是由“数据对象设计器”产生的解决方案时，界面设计器会自动加载该解决方案下的所有实体列表，并直接看到的就是上图。

如果我们是自己新建的解决方案时，则会出现下面的这个页面，并且我们需要自己来通过上图中的二个按钮，“选择程序集”或“选择模型文件”来加载实体对象。

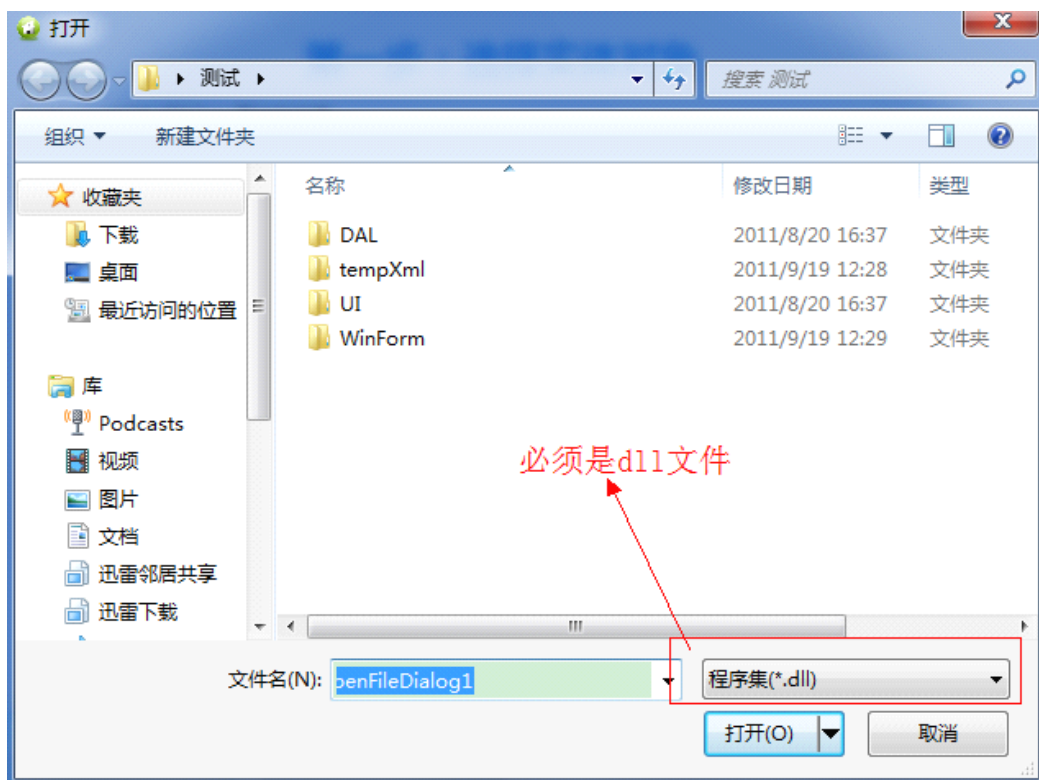
当我们是新建界面解决方案时，新建列表或窗体时，出现的界面如下：



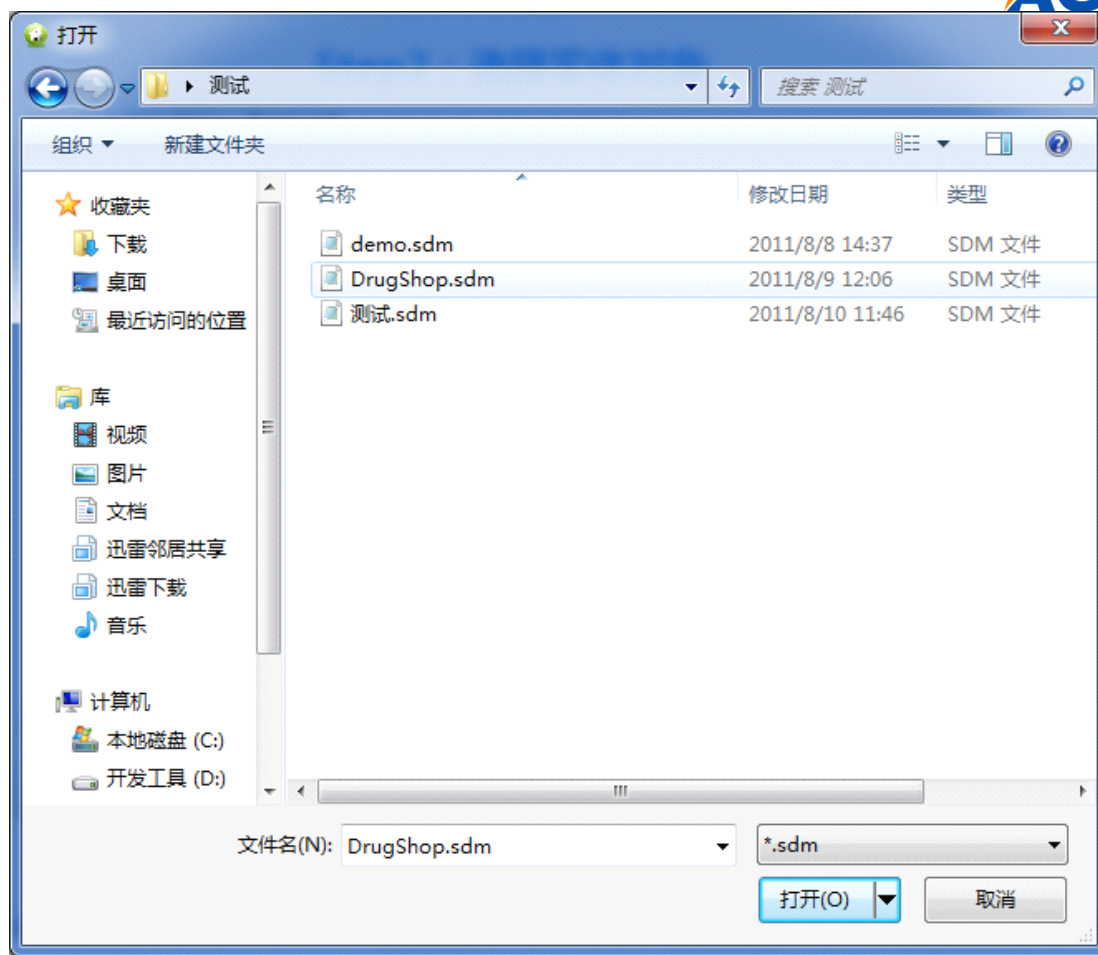
然后点击“选择模型文件”，模型文件请选择“数据对象设计器”的项目解决方案文件。



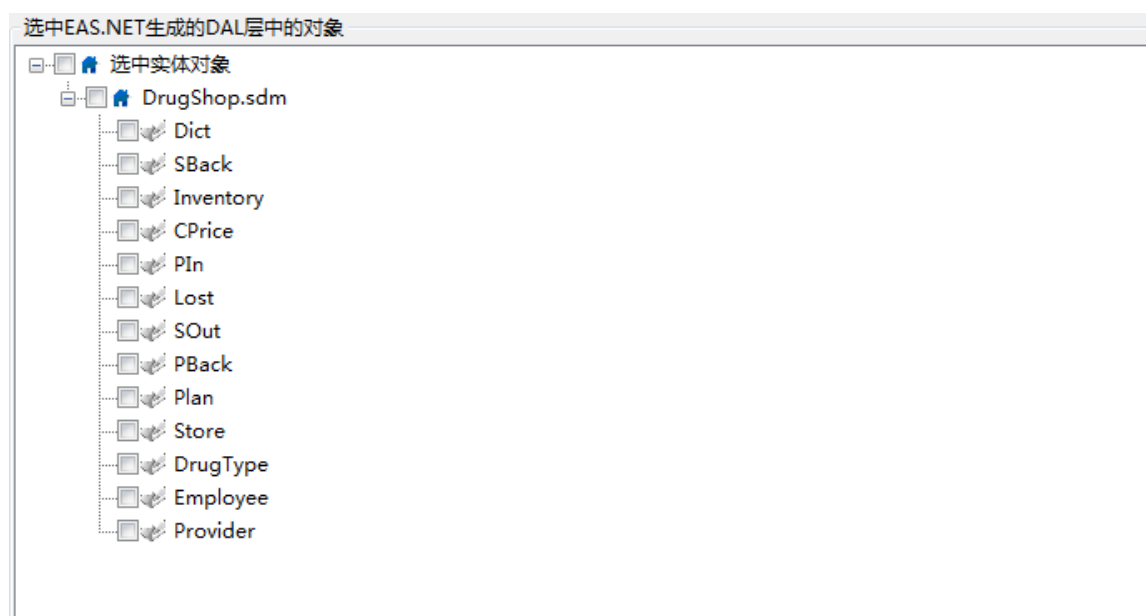
或者是点击“选择程序集”，则会加载该程序集下的所有的实体，当然前提该实体对象必须是 AgileEAS.NET 平台识别的实体对象才可以。



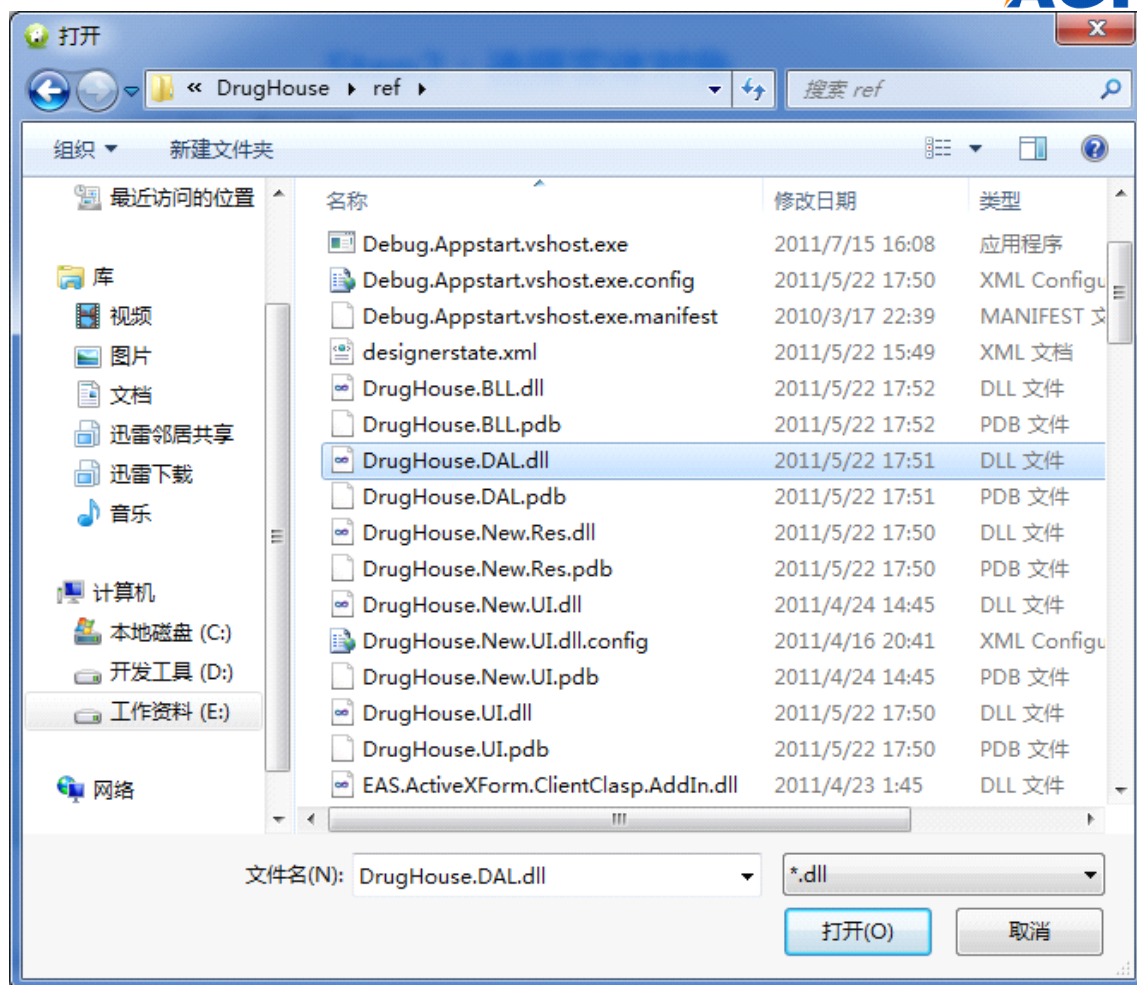
下面，我们来先来“选择模型文件”：



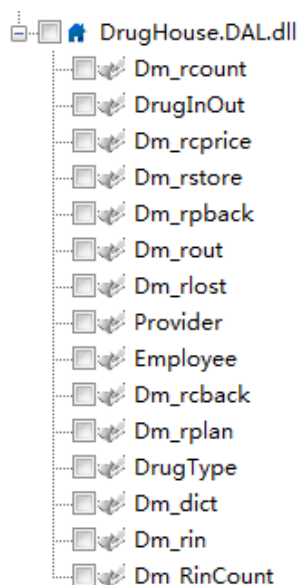
选择“.sdm”后缀名的文件。点击“确认”后：



该界面与上述的直接打开“数据对象设计器”项目解决方案的效果一样，下面我们来看看“选择程序集”弹出文件选择对话框：



注意请选择“.DAL”程序集，该程序集必须包含 AgileEAS.NET 平台规定的实体对象才能被识别，否则无法加载。



这样就完成了“实体对象”的加载和选择。

设置页面布局

当我们选择一个“实体对象”点击“下一步”后，出现如下界面。



第二步：设置界面布局

属性名	宽度	高度	列名	是否显示	是否查询项	排序
记录ID, 大于0的32位整数			Id	<input type="checkbox"/>	<input type="checkbox"/>	
Code			Code	<input type="checkbox"/>	<input type="checkbox"/>	
药品ID			Drugid	<input type="checkbox"/>	<input type="checkbox"/>	
中文名称			Cname	<input type="checkbox"/>	<input type="checkbox"/>	
药品规格			Spec	<input type="checkbox"/>	<input type="checkbox"/>	
大包装			Unit	<input type="checkbox"/>	<input type="checkbox"/>	
药品批发价格			Jobprice	<input type="checkbox"/>	<input type="checkbox"/>	
药品销售价格			Saleprice	<input type="checkbox"/>	<input type="checkbox"/>	
出库数量			Number	<input type="checkbox"/>	<input type="checkbox"/>	
生产厂商			Provider	<input type="checkbox"/>	<input type="checkbox"/>	
药品类型			Type	<input type="checkbox"/>	<input type="checkbox"/>	
药品效期			Timelimit	<input type="checkbox"/>	<input type="checkbox"/>	
出库日期			Eventtime	<input type="checkbox"/>	<input type="checkbox"/>	

基本信息： 命名空间: DrugShop 表单名称: SOutList

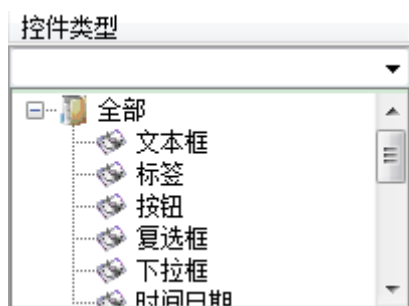
关联页面： 添加编辑关联的Form窗体: SOutEditor

其他设置： ☒ 启用添加 ☒ 启用编辑 ☒ 启用删除 ☒ 启用查询 ☒ 启用关闭 ☒ 启用打印

上图是“新建列表”的设置布局页面。

可以设置的属性及参数：

- 1、基本信息：设置窗体的命名空间及该界面名称。
- 2、管理页面：用于设置与该列表关联的 Form 窗体页面。(未提供)。
- 3、设置列表上的相关功能按钮。
- 4、设置 DataGridView 列表中的相关属性，设置控件的宽度与高度，设置是否为查询项(查询条件)和是否显示(是否在 GridView 上显示)。
- 5、设置控件的排序及控件类型。



新建窗体的设置布局页面则不同：

第二步：设置界面布局

属性名称	宽度	高度	是否新增项	是否显示	排序码	控件
记录ID，大于0的32位整数			<input type="checkbox"/>	<input type="checkbox"/>		
Code			<input type="checkbox"/>	<input type="checkbox"/>		
药品ID			<input type="checkbox"/>	<input type="checkbox"/>		
中文名称			<input type="checkbox"/>	<input type="checkbox"/>		
药品规格			<input type="checkbox"/>	<input type="checkbox"/>		
大包装			<input type="checkbox"/>	<input type="checkbox"/>		
药品批发价格			<input type="checkbox"/>	<input type="checkbox"/>		
药品销售价格			<input type="checkbox"/>	<input type="checkbox"/>		
出库数量			<input type="checkbox"/>	<input type="checkbox"/>		
生产厂商			<input type="checkbox"/>	<input type="checkbox"/>		
药品类型			<input type="checkbox"/>	<input type="checkbox"/>		
药品效期			<input type="checkbox"/>	<input type="checkbox"/>		
出库日期			<input type="checkbox"/>	<input type="checkbox"/>		
药品批号			<input type="checkbox"/>	<input type="checkbox"/>		

页面布局：☒ 单列多行 ☐ 多列多行 1 列 命名空间：DrugShop

其他设置：☒ 启用编辑 ☒ 启用保存 ☒ 启用关闭 表单名称：SOutEditor

上一步 下一步 取消

紧接着，设置页面的控件类型和控件的高度、宽度，是否新增项，是否显示。

界面设计器

第二步：设置界面布局

属性名称	宽度	高度	是否新增项	是否显示	排序码	控制
记录ID, 大于0的32位整数			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Code			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品ID			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
中文名称			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品规格			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
大包装			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品批发价格			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品销售价格			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
出库数量			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
生产厂商			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品类型			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品效期			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
出库日期			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
药品批号			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

☒ 全选(A)
☒ 全不选(U)

页面布局: ☒ 单列多行 ☐ 多列多行 1 列 命名空间: DrugShop
 其他设置: ☒ 启用编辑 ☒ 启用保存 ☒ 启用关闭 表单名称: SOutEditor

如果都不输入高度、宽度，则系统给予默认值。对于窗体的页面，支持页面布局的设置，比如多行多列，单列多行。

点击多列多行：

页面布局: ☐ 单列多行 ☒ 多列多行 1 列 命名空间: DrugShop
 其他设置: ☒ 启用编辑 ☒ 启用保存 ☒ 启用关闭 表单名称: SOutEditor

设置每行显示的列数。

其他设置“设置按钮”在界面上是否显示。

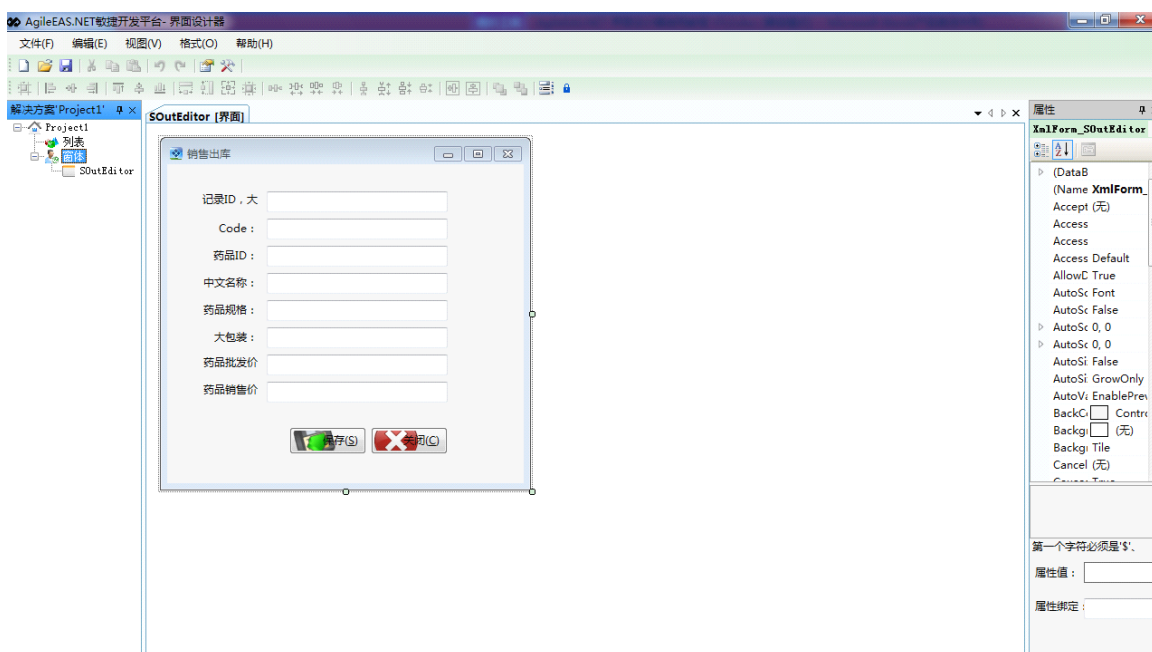
设置窗体文件的命名空间。

Datagridview 上支持全选与全不选的功能支持：

	宽度	高度	是否新增项	是否显示	排序码	控件类型
设员工的工作证...			<input type="checkbox"/>	<input type="checkbox"/>		
			<input type="checkbox"/>	<input type="checkbox"/>		
华人民共和国的...			<input type="checkbox"/>	<input type="checkbox"/>		
			<input type="checkbox"/>	<input type="checkbox"/>		
			<input type="checkbox"/>	<input type="checkbox"/>		
，所有部门中值...			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
，所有职务中值...			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
			<input type="checkbox"/>	<input type="checkbox"/>		
			<input type="checkbox"/>	<input type="checkbox"/>		
户士等，默认为0			<input type="checkbox"/>	<input type="checkbox"/>		
内权限级别，默...			<input type="checkbox"/>	<input type="checkbox"/>		

必须是单个列上选择。

设置完毕后，点击“下一步”则会系统会默认生成界面，在界面设计器中就能看到我们默认的输出的界面布局。



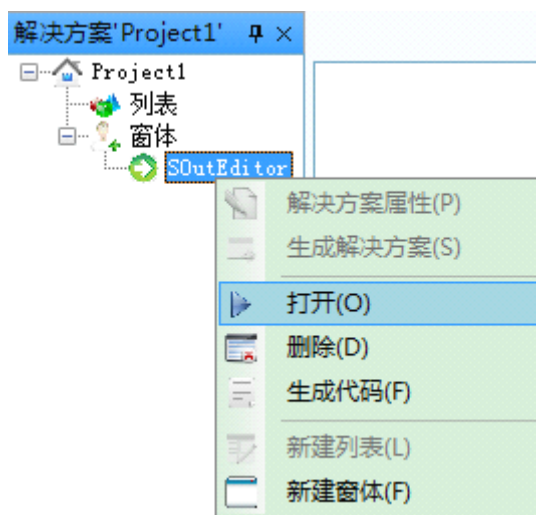
当然，如果我们界面上的控件不满意，或者是位置可以随意的拖拽，与 VS 的解决方案类似。

新增一个界面模型后，左侧的导航分组下的窗体分组会自动添加。此时我们可以保存解决方案。

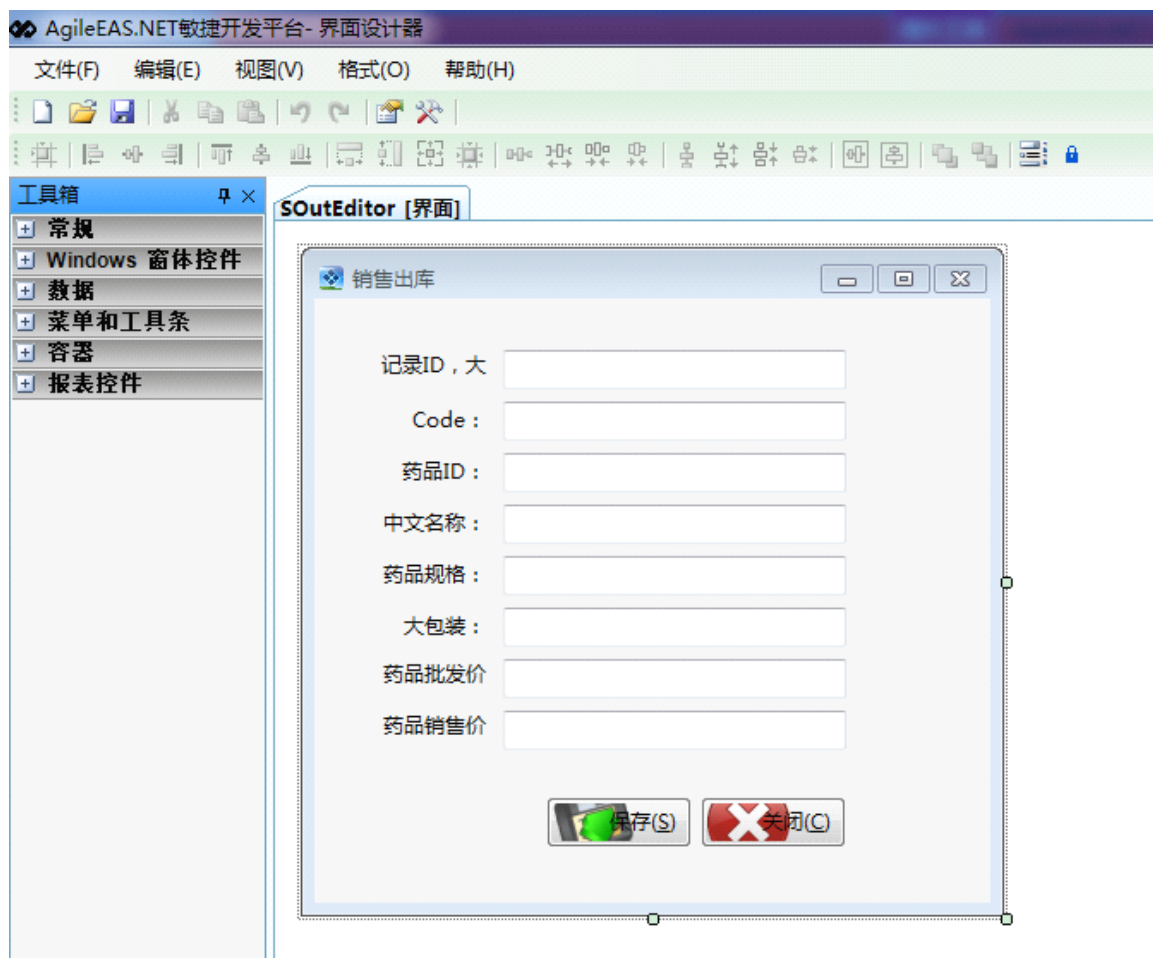
请选择“文件”菜单下的“保存解决方案”。

打开界面模型

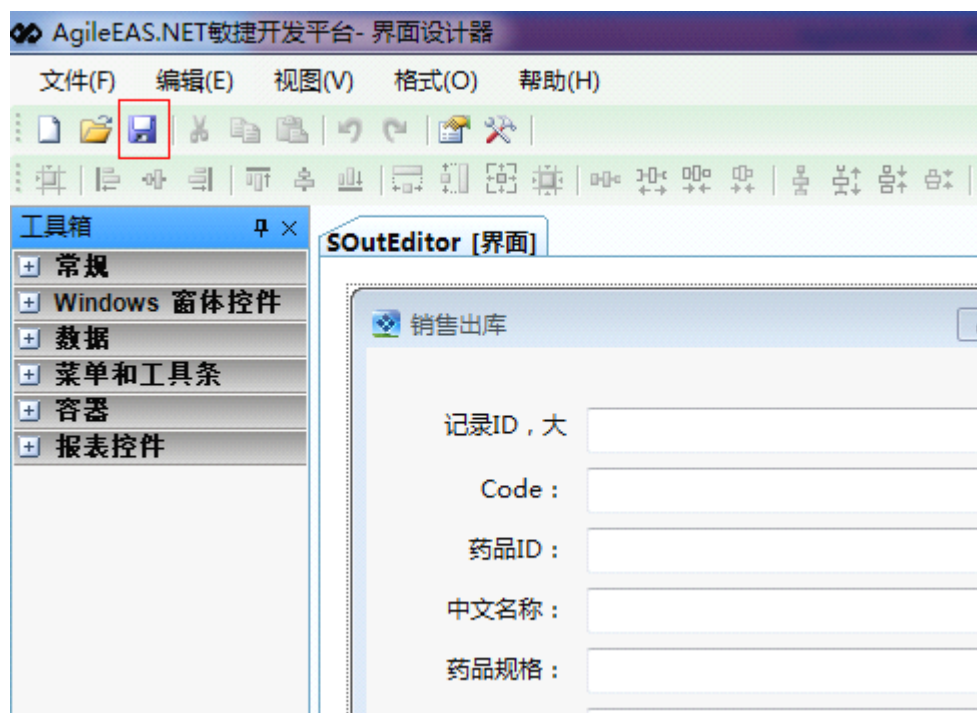
我们可以在导航栏的列表或者窗体分组内，选择界面模型节点，点击“右键”弹出快捷菜单中，出现如下菜单界面：



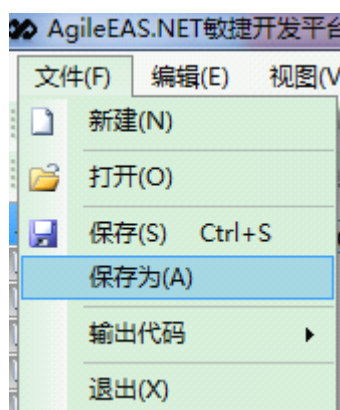
我们点击“打开”后，则会在右边的具体的内容区，显示该界面模型的真实运行界面。



左侧的工具箱中，我们可以拖拽控件到界面上，然后务必点击“保存”：



点击保存后，则会把所有的设计过的设计文件保存，或者是直接保存解决方案：



这样就完成界面设计过程。

关于界面布局

界面设计器的布局与 VS 开发中的一些对页面的布局对齐等功能是一样的，这里我就不详细的说明使用了。



大家就慢慢体会吧。

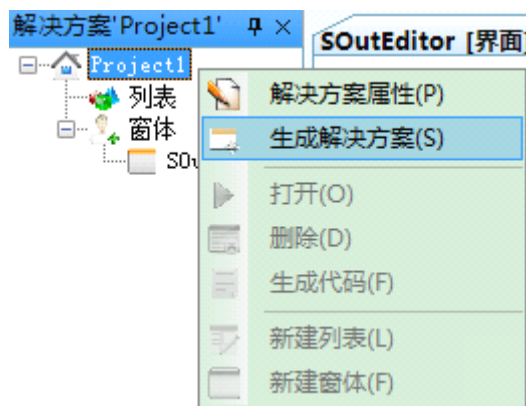
界面解决方案输出

概述

界面设计器提供基于解决方案的代码输出，可以一步生成供 vs2010 之上打开的数据层项目和解决方案。

输出代码

选择“解决方案名称” 点击右键：

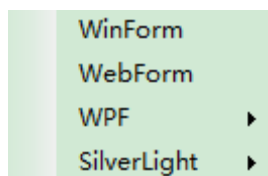


生成整个解决方案，点击“生成解决方案”时，出现配置界面：



输入解决方案的名称与解决方案的输出路径，点击“下一步”后则会开始生成代码文件与解决方案。

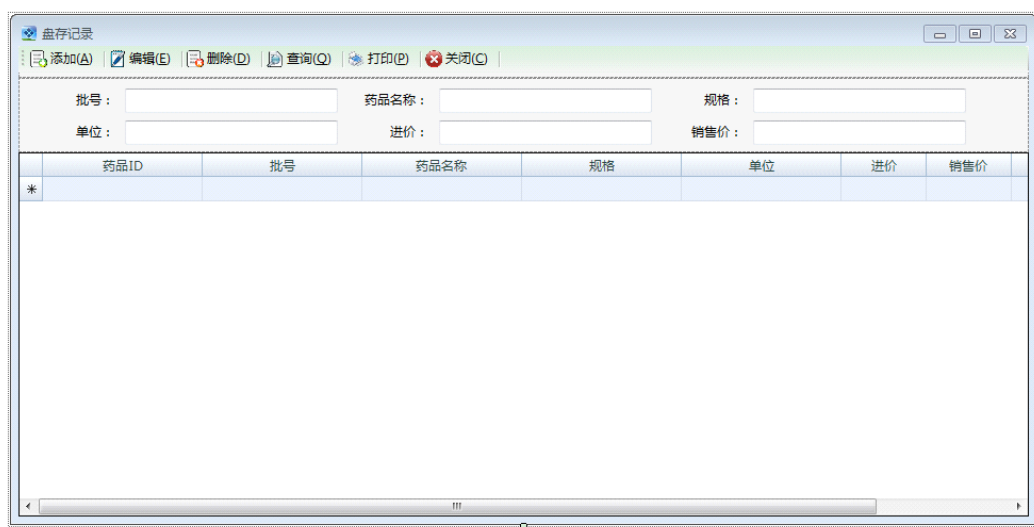
总体来说，代码输出提供如下四种方式：



下面就将这四种不同的输出来进行一一说明。

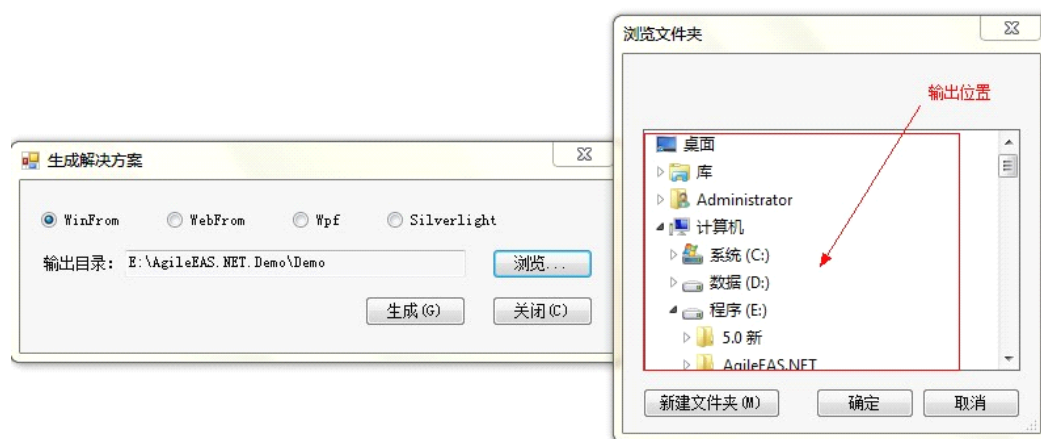
WinForm 界面代码输出

界面设计器生成的代码符合标准的 winform 界面的设计需求，系统会自动根据设计好的界面来生成相关的操作代码，比如列表上的查询代码：



布局上，上面是操作按钮，中间是查询区，下面是内容区。那么输出的 winform 的界面也与这个界面一样的形式。

我们点击输出解决方案，然后我们分别查看生成的列表与窗体的界面形式。














我们选择位置，然后设置解决方案名称。点击“下一步”后，在输出列表中显示如下内容：

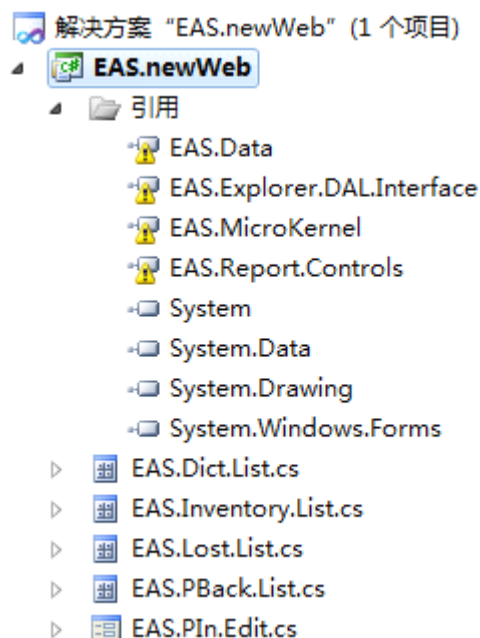
输出

设计器文件 C:\Users\hot\Desktop\WinForm\EAS.PIIn.Edit.Designer.cs 生成完毕
 原代码文件 C:\Users\hot\Desktop\WinForm\EAS.PIIn.Edit.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WinForm\EAS.Dict.List.Designer.cs 生成完毕
 原代码文件 C:\Users\hot\Desktop\WinForm\EAS.Dict.List.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WinForm\EAS.Inventory.List.Designer.cs 生成完毕
 原代码文件 C:\Users\hot\Desktop\WinForm\EAS.Inventory.List.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WinForm\EAS.Lost.List.Designer.cs 生成完毕
 原代码文件 C:\Users\hot\Desktop\WinForm\EAS.Lost.List.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WinForm\EAS.PBack.List.Designer.cs 生成完毕
 原代码文件 C:\Users\hot\Desktop\WinForm\EAS.PBack.List.cs 生成完毕

我们去指定的路径，查看，是否有解决方案被生成。输出的文件目录如下：

名称	修改日期	类型	大小
 EAS.Dict.List.cs	2011/8/10 12:58	Visual C# Sourc...	6 KB
 EAS.Dict.List.Designer.cs	2011/8/10 12:58	Visual C# Sourc...	21 KB
 EAS.Inventory.List.cs	2011/8/10 12:58	Visual C# Sourc...	6 KB
 EAS.Inventory.List.Designer.cs	2011/8/10 12:58	Visual C# Sourc...	21 KB
 EAS.Lost.List.cs	2011/8/10 12:58	Visual C# Sourc...	6 KB
 EAS.Lost.List.Designer.cs	2011/8/10 12:58	Visual C# Sourc...	19 KB
 EAS.newWeb.csproj	2011/8/10 12:58	Visual C# Projec...	5 KB
 EAS.PBack.List.cs	2011/8/10 12:58	Visual C# Sourc...	6 KB
 EAS.PBack.List.Designer.cs	2011/8/10 12:58	Visual C# Sourc...	21 KB
 EAS.PIIn.Edit.cs	2011/8/10 12:58	Visual C# Sourc...	6 KB
 EAS.PIIn.Edit.Designer.cs	2011/8/10 12:58	Visual C# Sourc...	10 KB

我们直接双击“.csproj”文件，用 vs2010 打开：



我们打开一个列表页面，查看设计器视图如下：

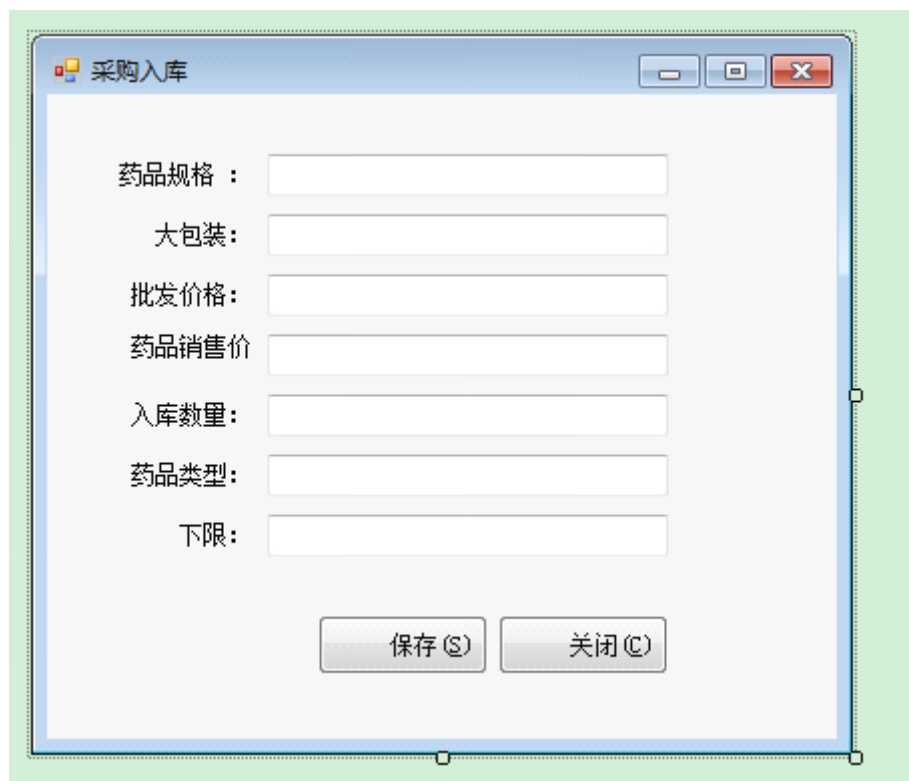


添加(A) | 编辑(E) | 删除(D) | 查询(Q) | 打印(P) | 关闭(C)

中文名称:	<input type="text"/>	药品规格:	<input type="text"/>	小包装:	<input type="text"/>
药品参考价	<input type="text"/>	药品库存上	<input type="text"/>	药品库存下	<input type="text"/>
药理类型:	<input type="text"/>				

与界面设计器中呈现的界面相同，就是没有图标资源了。

我们在来看看编辑页面：



采购入库

药品规格 :

大包装:

批发价格:

药品销售价

入库数量:

药品类型:

下限:

保存(S) 关闭(C)

与平时我们自己设置的界面相仿。

设计视图，我们看到了，下面我们来看看生成与界面设计器配套的代码文件。

1、 列表

```
using EAS.Data.Linq;

using DrugShop;
using DrugShop.DAL;
using Project1;

namespace DrugShop
{
    [Module("e3582842-91e7-4733-8978-7c0480c23317", "PInList", "PInList")]
    public partial class PInList : UserControl
    {
        private IList<PIn> m_DataList = null;

        public PInList()
        {
            InitializeComponent();
            this.dataGridView1.DataSource = this.datasourcedataGridView1;
            this.m_DataList = new List<PIn>();
        }

        [ModuleStart]
        public void StartEx()
        {
        }
    }
}
```

系统会默认注册模块的相关信息。并且会输出初始化方法。

```
private void tsbAdd_Click(object sender, EventArgs e)
{
    PInEditor editor=new PInEditor();
    editor.ShowDialog();
}

private void tsbEdit_Click(object sender, EventArgs e)
{
    PInEditor editor=new PInEditor();
    editor.ShowDialog();
}

private void tsbDelete_Click(object sender, EventArgs e)
{
    PIn selectEntity= this.dataGridView1.CurrentCell.OwningRow.DataBoundItem as PIn;
    if(selectEntity==null)
    {
        return;
    }
    this.dataGridView1.Rows.Remove(this.dataGridView1.CurrentCell.OwningRow) ;
    if (this.m_DataList.Contains(selectEntity))
        this.m_DataList.Remove(selectEntity);
    selectEntity.Delete();
}
```

系统会默认生成界面相关的操作方法，当我们选择列表之上的窗体文件的时候，会默认在 tsbAdd 与 tsbEdit 方法内添加与编辑页面关联起来。

2、窗体

```
namespace DrugShop
{
    /// <summary>
    /// 销售出库。
    /// </summary>
    public partial class SOutEditor : Form
    {
        public SOutEditor()
        {
            InitializeComponent();
        }
        private SOut m_DataEntity;
        public SOut DataEntity
        {
            get
            {
                return this.m_DataEntity;
            }
            set
            {
                if(value == null)
                    return;
                this.m_DataEntity = value;
                this.RefreshData();
            }
        }
    }
}
```

默认会生成该窗体内的当前编辑或者添加的实体对象绑定。

```
private void RefreshData()
{
    if(this.DataEntity != null)
    {
        this.TextBox_ID.Text=this.m_DataEntity.Id.ToString();
        this.TextBox_CODE.Text=this.m_DataEntity.Code;
        this.TextBox_DRUGID.Text=this.m_DataEntity.Drugid;
        this.TextBox_CNAME.Text=this.m_DataEntity.Cname;
        this.TextBox_SPEC.Text=this.m_DataEntity.Spec;
        this.TextBox_UNIT.Text=this.m_DataEntity.Unit;
        this.TextBox_JOBPRICE.Text=this.m_DataEntity.Jobprice.ToString();
        this.TextBox_SALEPRICE.Text=this.m_DataEntity.Saleprice.ToString();
    }
}
```

保存添加的方法，还有就是保存之前的数据验证方法，包括关闭窗体的方法。

验证方法：

```
private bool ValidInput()
{
    if ( this.TextBox_ID.Text == "" )
    {
        MessageBox.Show(this, "记录ID , 大于0的32位整数 : 必填项,不能为空!", "错误", MessageBoxButtons.OK, MessageBoxIcon.Error);
        this.TextBox_ID.Focus();
        return false;
    }
    if ( this.TextBox_CODE.Text == "" )
    {
        MessageBox.Show(this, "Code : 必填项,不能为空!", "错误", MessageBoxButtons.OK, MessageBoxIcon.Error);
        this.TextBox_CODE.Focus();
        return false;
    }
    if ( this.TextBox_DRUGID.Text == "" )
    {
        MessageBox.Show(this, "药品ID : 必填项,不能为空!", "错误", MessageBoxButtons.OK, MessageBoxIcon.Error);
        this.TextBox_DRUGID.Focus();
        return false;
    }
    if ( this.TextBox_CNAME.Text == "" )
    {
        MessageBox.Show(this, "中文名称 : 必填项,不能为空!", "错误", MessageBoxButtons.OK, MessageBoxIcon.Error);
        this.TextBox_CNAME.Focus();
        return false;
    }
}
```

保存方法:

```
//基本信息
dataEntity.Id = this.TextBox_ID.Text != "" ? Convert.ToInt32(this.TextBox_ID.Text) : 0;
dataEntity.Code = this.TextBox_CODE.Text;
dataEntity.Drugid = this.TextBox_DRUGID.Text;
dataEntity.Cname = this.TextBox_CNAME.Text;
dataEntity.Spec = this.TextBox_SPEC.Text;
dataEntity.Unit = this.TextBox_UNIT.Text;
dataEntity.Jobprice = this.TextBox_JOBPRICE.Text != "" ? Convert.ToDecimal(this.TextBox_JOBPRICE.Text) : 0;
dataEntity.Saleprice = this.TextBox_SALEPRICE.Text != "" ? Convert.ToDecimal(this.TextBox_SALEPRICE.Text) : 0;

if (this.DataEntity == null)
{
    dataEntity.Insert();
}
else
{
    dataEntity.Update();
}

this.m_DataEntity = dataEntity;
```













Web 解决方案输出

Web 解决方案的输出与之前的 Winfrom 的解决方案的输出类似, 唯一不同的是, 我们在输出代码的时候, 多输出了一个样式表文件, 该文件用于设置输出的 web 页面的样式与布局。

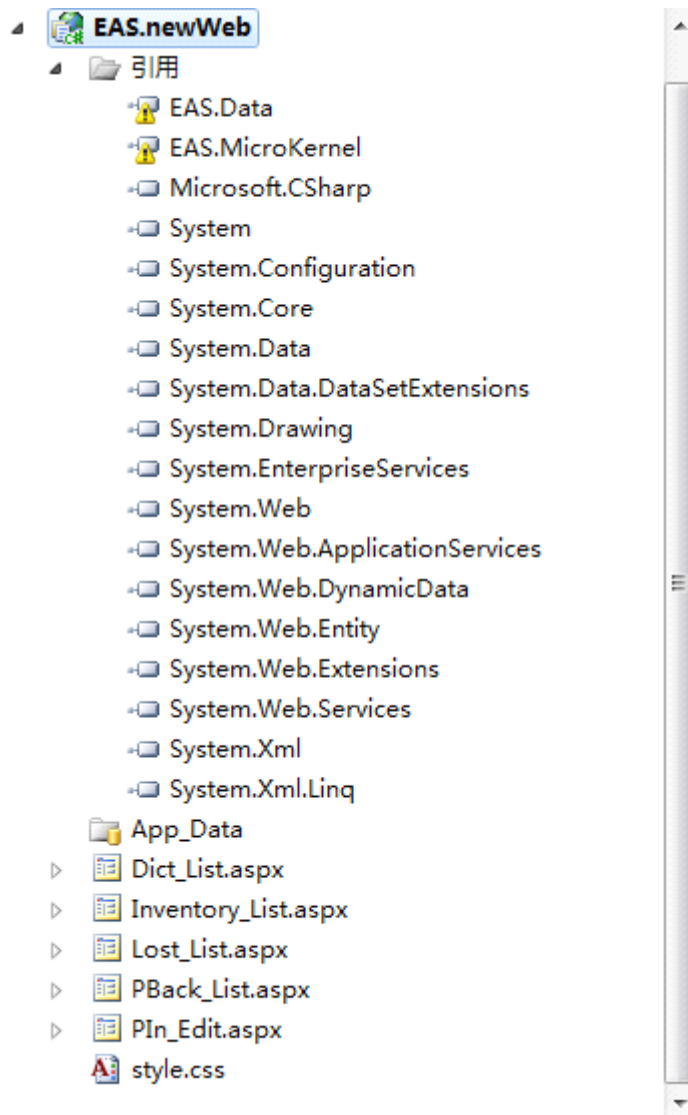
“生成解决方案”时选择“Web”:



设置“解决方案名称”，设置解决方案的输出位置，点击“下一步”。生成的 Web 文件列表如下：

名称	修改日期	类型	大小
 Dict_List.aspx	2011/8/10 14:27	ASP.NET Server ...	6 KB
 Dict_List.aspx.cs	2011/8/10 14:27	Visual C# Sourc...	6 KB
 EAS.newWeb.csproj	2011/8/10 14:27	Visual C# Projec...	5 KB
 Inventory_List.aspx	2011/8/10 14:27	ASP.NET Server ...	6 KB
 Inventory_List.aspx.cs	2011/8/10 14:27	Visual C# Sourc...	6 KB
 Lost_List.aspx	2011/8/10 14:27	ASP.NET Server ...	6 KB
 Lost_List.aspx.cs	2011/8/10 14:27	Visual C# Sourc...	5 KB
 PBack_List.aspx	2011/8/10 14:27	ASP.NET Server ...	6 KB
 PBack_List.aspx.cs	2011/8/10 14:27	Visual C# Sourc...	5 KB
 PIn_Edit.aspx	2011/8/10 14:27	ASP.NET Server ...	3 KB
 PIn_Edit.aspx.cs	2011/8/10 14:27	Visual C# Sourc...	6 KB
 style.css	2011/8/10 14:27	CSS Document	6 KB

直接打开“.csproj”文件，打开后的效果如下：



查看生成的界面效果：

1、 列表

盘存记录											
批号:	<input type="text"/>	药品名称:	<input type="text"/>	规格:	<input type="text"/>						
单位:	<input type="text"/>	进价:	<input type="text"/>	销售价:	<input type="text"/>						
查 询											
药品ID	批号	药品名称	规格	单位	进价	销售价	库存	实存	选择	编辑	删除
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	选择	编辑	删除
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	选择	编辑	删除
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	选择	编辑	删除
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	选择	编辑	删除
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	选择	编辑	删除

下面还有操作按钮

数据绑定	首页 上一面 下一面 尾页 到第 <input type="text"/> 页 <input type="button" value="GO"/>
全选 全不选 编辑 删除	

我们再来看看，后台的代码：

```
using EAS.Data.ORM;
using EAS.Data.Access;
using EAS.Modularization;
using EAS.Services;
using EAS.Data.Linq;

namespace DrugShop
{
    public partial class Dict_List : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!this.IsPostBack)
            {
                //页面GridView绑定
                this.GridViewBind();
            }
        }

        public EAS.Data.Access.IDataAccessor DataAccessor
        {
            get
            {
                return EAS.Context.ContextHelper.GetContext().Container.GetComponentInstance("DataAccessor") as EAS.Data.Access.I
            }
        }
    }
}

GridView操作
```

其他的更多的操作代码：

```
protected void dataGridView1_PageIndexChanging(object sender, GridViewPageEventArgs e) {  
  
protected void dataGridView1_RowCommand(object sender, GridViewCommandEventArgs e) {  
  
[查询数据]
```

2、窗体

采购入库	
药品规格：	<input type="text"/>
大包装：	<input type="text"/>
批发价格：	<input type="text"/>
药品销售价格：	<input type="text"/>
入库数量：	<input type="text"/>
药品类型：	<input type="text"/>
下限：	<input type="text"/>
保存 (AS) 关闭 (AC)	

下面我们来看看后台生成的代码：

```
using EAS.Data;  
using EAS.Data.ORM;  
using EAS.Data.Access;  
using EAS.Modularization;  
using EAS.Services;  
using EAS.Data.Linq;  
  
namespace DrugShop  
{  
    public partial class PIn_Edit : System.Web.UI.Page  
    {  
        protected void Page_Load(object sender, EventArgs e)  
        {  
            if (!this.IsPostBack)  
            {  
                //页面GridView绑定  
                int id= this.Request.QueryString["ID"] == null ? -1 : int.Parse(this.Request.QueryString["ID"]);  
                this.InitInfo(id);  
            }  
        }  
        public void InitInfo(int id)  
        {  
            if(id>0)  
            {  

```

其他方法：

```
private DrugShop.PIn currentDataEntity;
public DrugShop.PIn CurrentDataEntity
{
    get
    {
        return this.currentDataEntity;
    }
    set
    {
        if(value == null)
            return;
        this.currentDataEntity = value;
        this.InitBaseInfo();
    }
}
private void InitBaseInfo() ...

private bool ValidInput() ...

/// <summary>
/// 【确定】更新。
/// </summary>
private void btnOK_Click(object sender, System.EventArgs e) ...
```

上面就是 web 下的代码输出。

WPF 解决方案输出

在生成解决方案的界面上，选择“输出 WPF”














输出的生成信息如下：

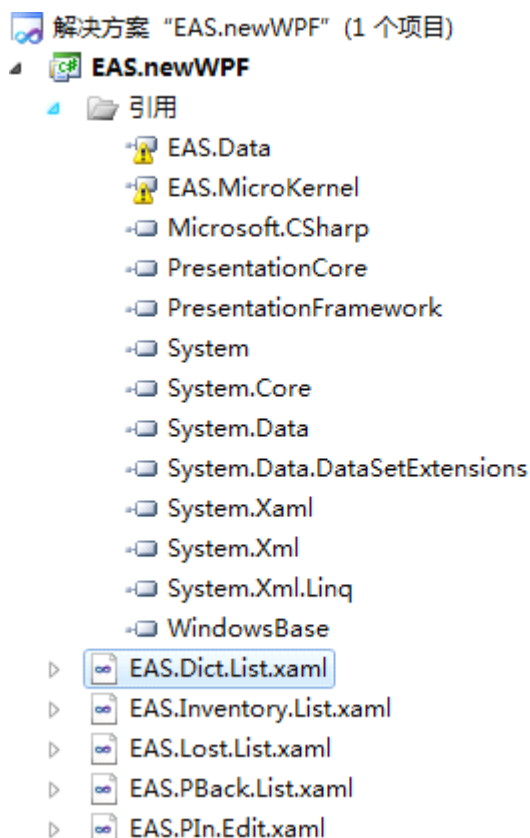
输出

设计器文件 C:\Users\hot\Desktop\WPF\EAS.PIn.Edit.xaml 生成完毕
 源代码文件 C:\Users\hot\Desktop\WPF\EAS.PIn.Edit.xaml.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WPF\EAS.Dict.List.xaml 生成完毕
 源代码文件 C:\Users\hot\Desktop\WPF\EAS.Dict.List.xaml.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WPF\EAS.Inventory.List.xaml 生成完毕
 源代码文件 C:\Users\hot\Desktop\WPF\EAS.Inventory.List.xaml.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WPF\EAS.Lost.List.xaml 生成完毕
 源代码文件 C:\Users\hot\Desktop\WPF\EAS.Lost.List.xaml.cs 生成完毕
 设计器文件 C:\Users\hot\Desktop\WPF\EAS.PBack.List.xaml 生成完毕
 源代码文件 C:\Users\hot\Desktop\WPF\EAS.PBack.List.xaml.cs 生成完毕

生成的 WPF 文件结构如下：

名称	修改日期	类型	大小
 EAS.Dict.List.xaml	2011/8/10 15:24	XAML 文件	4 KB
 EAS.Dict.List.xaml.cs	2011/8/10 15:24	Visual C# Sourc...	3 KB
 EAS.Inventory.List.xaml	2011/8/10 15:24	XAML 文件	4 KB
 EAS.Inventory.List.xaml.cs	2011/8/10 15:24	Visual C# Sourc...	3 KB
 EAS.Lost.List.xaml	2011/8/10 15:24	XAML 文件	3 KB
 EAS.Lost.List.xaml.cs	2011/8/10 15:24	Visual C# Sourc...	3 KB
 EAS.newWPF.csproj	2011/8/10 15:24	Visual C# Projec...	5 KB
 EAS.PBack.List.xaml	2011/8/10 15:24	XAML 文件	4 KB
 EAS.PBack.List.xaml.cs	2011/8/10 15:24	Visual C# Sourc...	3 KB
 EAS.PIn.Edit.xaml	2011/8/10 15:24	XAML 文件	3 KB
 EAS.PIn.Edit.xaml.cs	2011/8/10 15:24	Visual C# Sourc...	6 KB

我们双击“.csproj”项目文件打开。



添加相关的引用。生成的列表与窗体页面如下：

1、列表

我们打开一个列表页面：



后台代码与之前的 winform 的页面一样，wpf 列表也是作为一个插件，默认生成插件相关的标记。

```
using EAS.Data;
using EAS.Data.ORM;
using EAS.Data.Access;
using EAS.Modularization;
using EAS.Services;
using EAS.Data.Linq;

namespace DrugShop
{
    [Module("455934b9-73a3-466e-ae03-f55d45e7f8e1", "Dict.List", "Dict.List")]
    public partial class Dict_List : UserControl
    {
        public Dict_List()
        {
            InitializeComponent();

            [ModuleStart]
            public void StartEx()
            {
                this.InitInfo();
            }

            private IList<DrugShop.Dict> dictList = null;
        }
    }
}
```

其他操作代码:

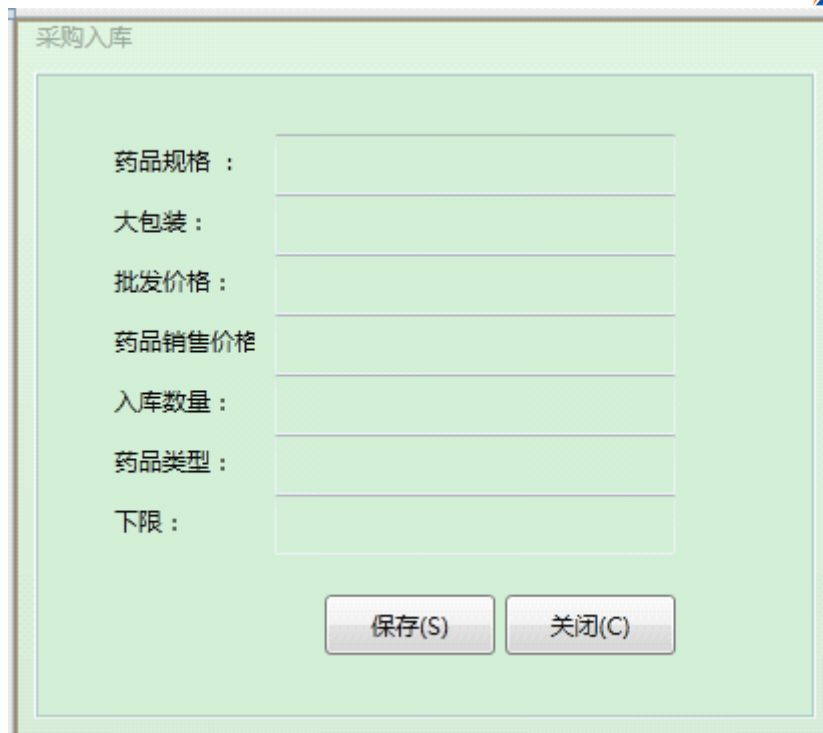
```
private void TsbAdd_Click(object sender, EventArgs e) { ... }
private void TsbEdit_Click(object sender, EventArgs e) { ... }
private void TsbDelete_Click(object sender, EventArgs e) { ... }
private void TsbQuery_Click(object sender, EventArgs e) { ... }
private void TsbPrint_Click(object sender, EventArgs e) { ... }
private void TsbClose_Click(object sender, EventArgs e) { ... }
```

查询数据

上述是工具栏中的相关按钮事件。

2、窗体

先看看窗体的界面效果。



相关代码:


```
using EAS.Data;
using EAS.Data.ORM;
using EAS.Data.Access;
using EAS.Modularization;
using EAS.Services;
using EAS.Data.Linq;


namespace DrugShop
{
    /// <summary>
    /// 采购入库
    /// </summary>
    public partial class PIn_Edit : Window
    {
        public PIn_Edit()
        {
            InitializeComponent();
        }

        private DrugShop.PIn currentDataEntity;
        public DrugShop.PIn CurrentDataEntity
        {
            get
            {
                return this.currentDataEntity;
            }
        }
    }
}
```

更多代码，初始化绑定，信息验证，信息保存等：


```
private void InitBaseInfo()
{
    if(this.currentDataEntity != null)
    {
        this.TextBox_SPEC.Text=this.currentDataEntity.Spec;
        this.TextBox_UNIT.Text=this.currentDataEntity.Unit;
        this.TextBox_JOBPRICE.Text=this.currentDataEntity.Jobprice.ToString();
        this.TextBox_SALEPRICE.Text=this.currentDataEntity.Saleprice.ToString();
        this.TextBox_NUMBER.Text=this.currentDataEntity.Number.ToString();
        this.TextBox_TYPE.Text=this.currentDataEntity.Type.ToString();
        this.TextBox_DOWNLIMIT.Text=this.currentDataEntity.Downlimit.ToString();
    }
}

private bool ValidInput() 

/// <summary>
/// 【确定】更新。
/// </summary>
private void btnOK_Click(object sender, System.EventArgs e) 
```

上面就完成了 WPF 解决方案及代码的生成。

结束语

在市场激烈的今天，软件企业面临着极多种多样的挑战，如果在市场快速变化的情况下脱颖而出，如果能够对市场变化及时做出反应，以较低成本推出市场所需要的产品并持续改进产品成为成功的必要。

企业之间的竞争很大程度上是成本的竞争，AgileEAS.NET 平台以及敏捷并行开发方法实践能大大够缩短软件产品开发周期，降低软件产品的开发、实施和维护成本，能够很大程度上提升软件企业的争力。

敏捷软件工程实验室秉承“敏捷反应，快速适应”的宗旨，始终如一的对中小软件企业提供 AgileEAS.NET 平台技术支持、升级服务，为软件企业提供先进的快速开发平台，同时也提供 Microsoft .Net 开发技术培训、技术管理咨询服务，帮助软件企业在激烈的市场竞争中取得不断的成功。

联系我们

我为完善、改进和推广 AgileEAS.NET 而成立了敏捷软件工程实验室，是一家研究、推广和发展新技术，并致力于提供具有自主知识产权的业务基础平台软件，以及基于业务基础平台开发的管理软件的专业软件提供商。主要业务是为客户提供软件企业研发管理解决方案、企业管理软件开发，以及相关的技术支持，管理及技术咨询与培训业务。

AgileEAS.NET 平台自 2004 年秋呱呱落地一来，我就一直在逐步完善和改进，也被应用于保险、医疗、电子商务、房地产、铁路、教育等多个应用，但一直都是以我个人在推广，2010 年因为我辞职休息，我就想到把 AgileEAS.NET 推向市场，让更多的人使用。

下面我简单介绍一下我吧，魏琼东，系统分析师、高级项目经理、软件设计师，CSAI 顾问团顾问。有近 10 年开发和管理经验，在多家 IT 企业从事过架构师、项目经理、开发部经理、技术总监、总工程师，为医疗、保险、教育、农业、房地产、电子政务等行业研发软件。有着丰富的理论知识及资深的行业软件经验，擅长企业软件过程改进、系统架构分析与设计、.NET 平台框架技术，SQL Server/ORACLE 数据库技术、分布式架构体系，尤其对中小软件企业软件研发管理体系有着深入的理解与应用，目前主要致力于中小型企业一体化信息化建设方案，为中小型企业提供新形势下的信息化建设方案、企业信息化管理咨询，企业信息化流程改进等服务。

我的技术团队成员都是合作多年的老朋友，因为这个平台是免费的，所以也没有什么收入，都是由程序员的那种理想与信念坚持，在此我感谢一起奋斗的朋友。

团队网站: <http://www.agilelab.cn>,

AgileEAS.NET 网站: <http://www.smart eas.net>

官方博客:<http://eastjade.cnblogs.com>

QQ:47920381,AgileEAS.NET

QQ 群: 113723486 (AgileEAS SOA 平台) /上限 1000 人

199463175 (AgileEAS SOA 交流) /上限 1000 人

120661978 (AgileEAS.NET 平台交流) /上限 1000 人

212867943 (AgileEAS.NET 研究) /上限 500 人

147168308 (AgileEAS.NET 应用) /上限 500 人

172060626 (深度 AgileEAS.NET 平台) /上限 500 人

116773358 (AgileEAS.NET 平台) /上限 500 人

125643764 (AgileEAS.NET 探讨) /上限 500 人

193486983 (AgileEAS.NET 平台) /上限 500 人

邮件:james@agilelab.cn,mail.james@qq.com,

电话: 18629261335。