NHibernate 开发架构开发文档

作者：yangxie（谢炀）

文档最后更新：2008-05-03

文档版本号：1.0.0

1 概述

1.1 简介

本开发框架是一个集成了 Spring.NET & NHibernate 的 WEB 快速开发框架，包括框架基础库以及与之配套的代码生成工具。

框架使用开源组件 NHibernate 2.0.1 访问数据，使用 Spring.Net 1.2.0 做为容器框架管理数据层、业务层、表现层之间的耦合关系。

框架使用到以下第三方组件：

- antlr.runtime.dll
- Castle.Core.dll
- Castle.DynamicProxy2.dll
- Common.Logging.dll
- Common.Logging.Log4Net.dll
- Iesi.Collections.dll
- log4net.dll
- NHibernate.dll
- Spring.Aop.dll
- Spring.Core.dll
- Spring.Data.dll
- Spring.Data.NHibernate20.dll
- Spring.Web.dll

框架包含两个配套的代码生成工具分别用于辅助生成后台代码以及表现层页面。

框架功能：

1 支持在类的方法上面打标签控制事务。
2 支持类似 Linq 的动态条件构造查询。
3 配套的代码生成工具可以自动根据数据库表结构生辰所有的业务层、数据层类。

2 创建一个入门项目

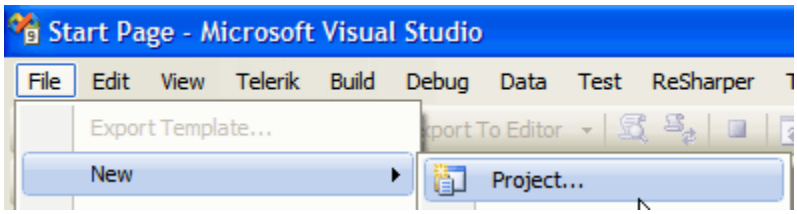
下面创建一个基于微软 Northwind 的数据库操作例子来简要的介绍一下如何使用框架。

2.1 创建数据库

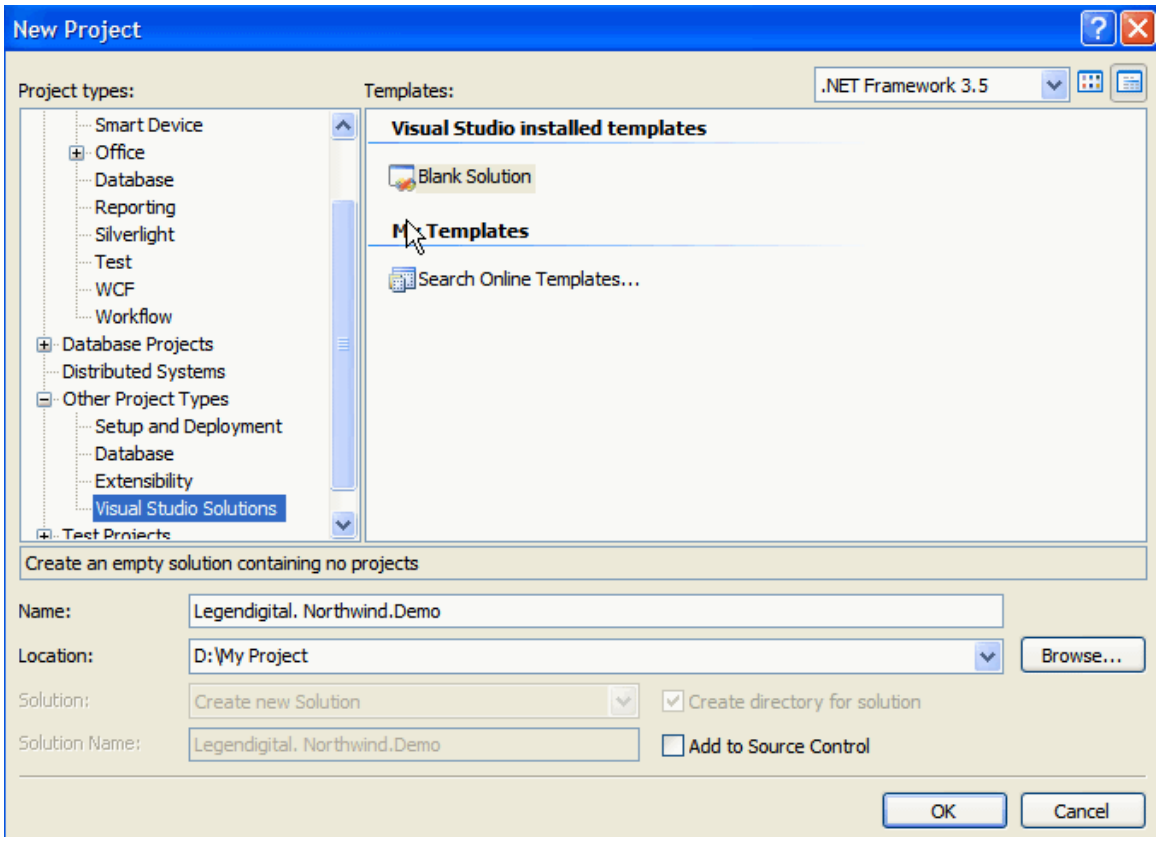
安装 Northwind 数据库，附加 Northwind 到数据库

2.1.1 创建示例项目

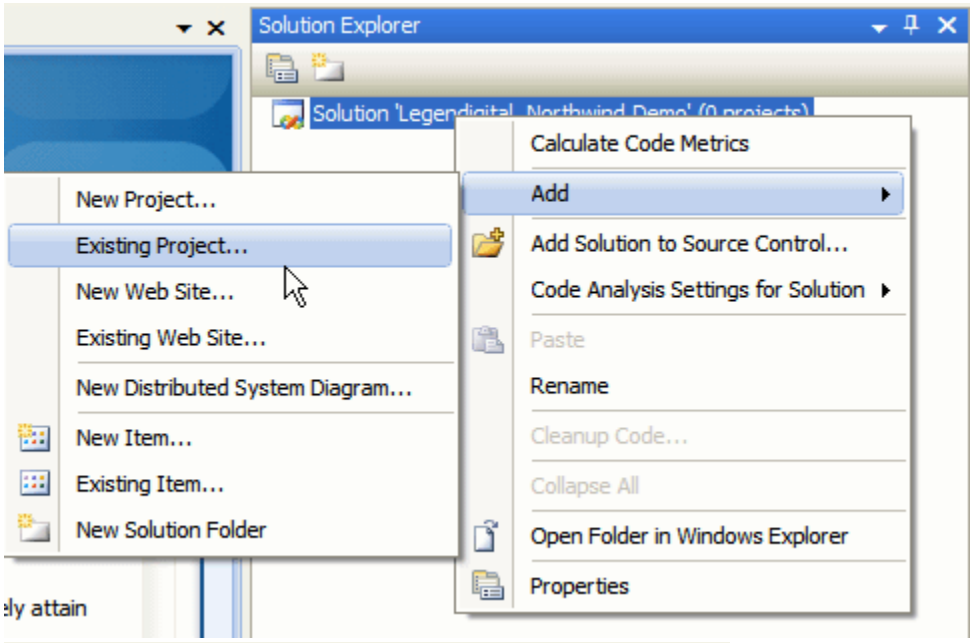
1. 创建一个空的 Legendigital. Northwind.Demo 解决方案
   - 点击菜单”File->New->Project”

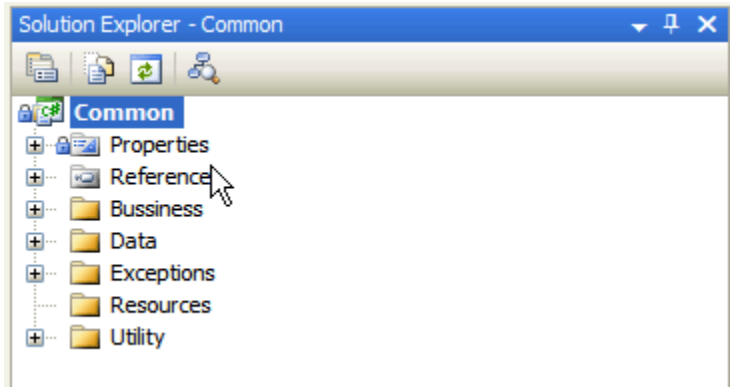


   - 选择”Other Project Type->Visual Stuido Solution-> Blank Solution”

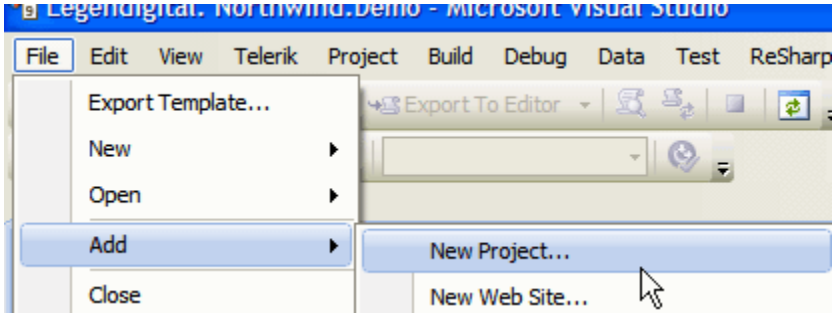


2. 添加 Legendigital.Framework.Common 项目到解决方案
   - 添加框架基础库项目引用到解决方案

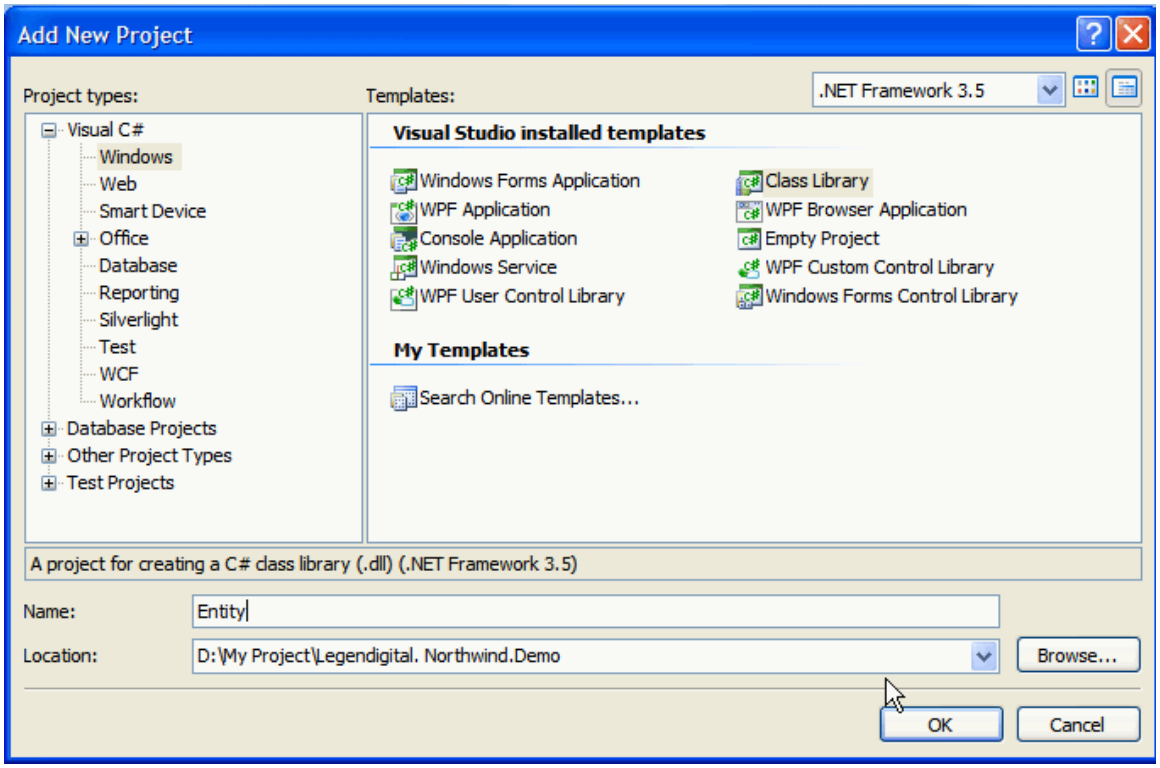




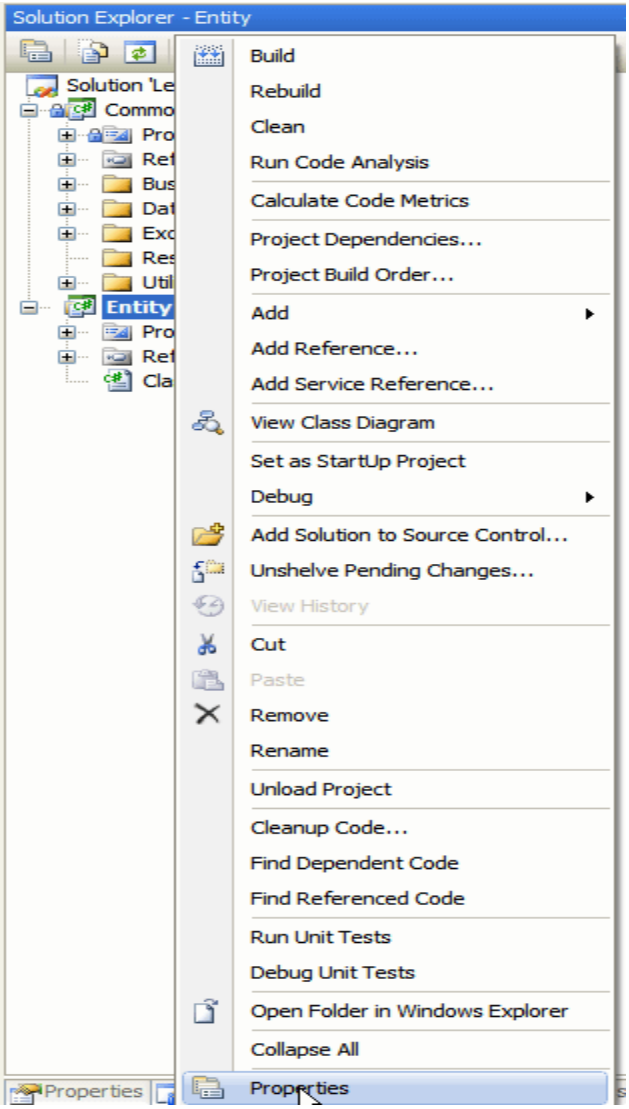
3. 创建实体层项目 Legendigital. Northwind.Entity
   接下来需要创建实体层项目
   - 点击菜单”File->Add->New Project”



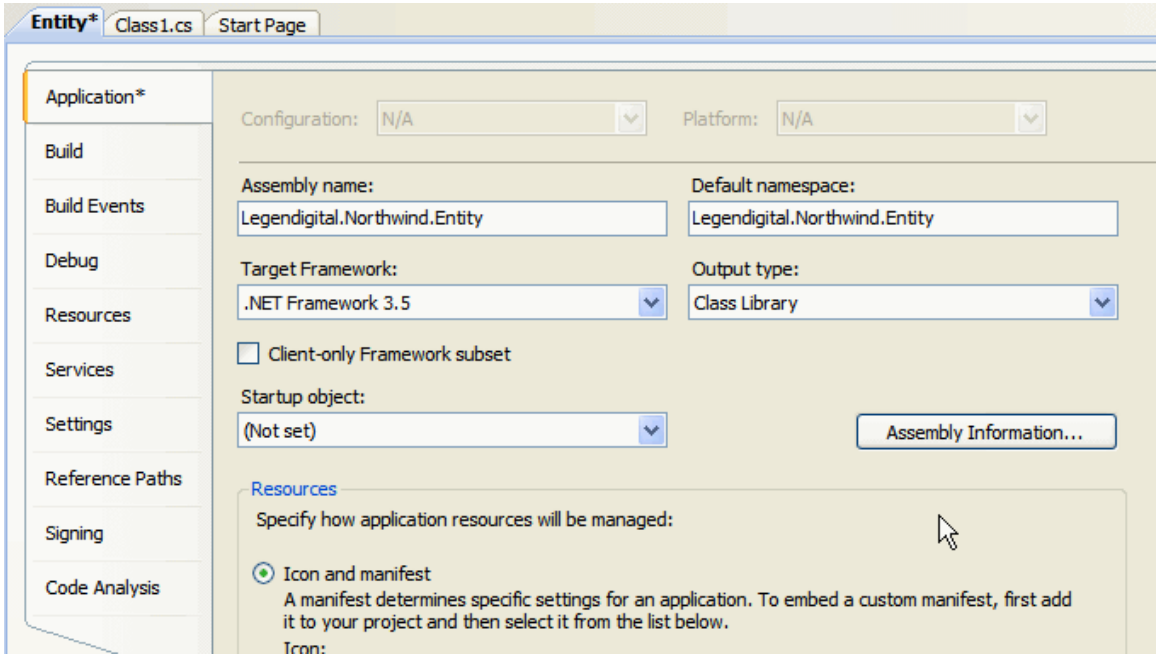
- 创建名为 Entity 的 Class Library 项目



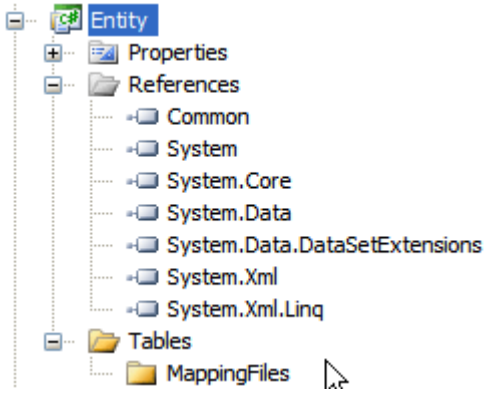
- 在解决方面窗口里面选择 Entity 项目右键点击选择属性，打开项目属性设置窗口。



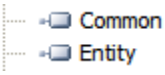
- 修改默认名称空间名和默认程序集名修改为 Legendigital. Northwind.Entity。按 Ctrl+S 进行保存。



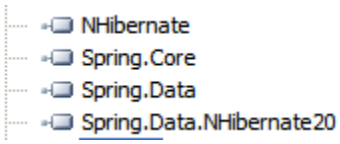
- Entity 项目添加对 Common 项目的引用。
- 删除自动生成的 class1.cs
- 在项目里面创建 Tables—> MappingFiles 目录



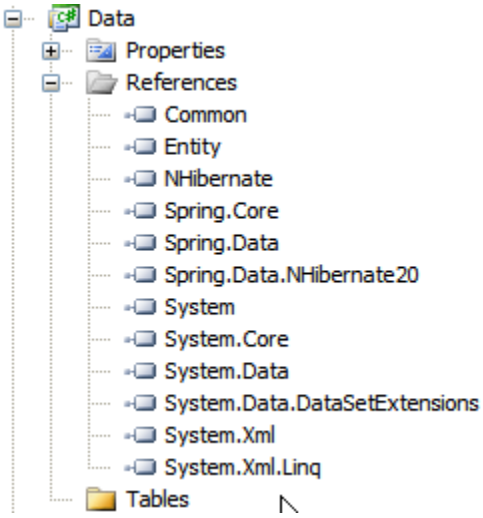
4. 创建数据层项目 Legendigital. Northwind.Data
    - 点击菜单”File->Add->New Project”
    - 创建名为 Data 的 Class Library 项目
    - 在解决方面窗口里面选择 Data 项目右键点击选择属性，打开项目属性设置窗口。
    - 修改默认名称空间名和默认程序集名修改为 Legendigital.Northwind.Data。按 Ctrl+S 进行保存。
    - Data 项目添加对 Common 项目以及 Entity 项目的引用。



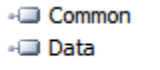
- 确保以下第三方 Dll 被 Data 项目引用。



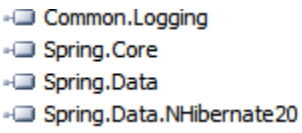
- 删除自动生成的 class1.cs
- 在项目中创建 Tables 目录



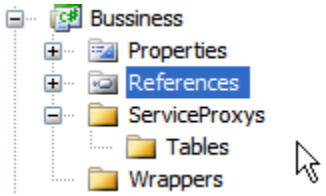
5. 创建业务层项目 Legendigital. Northwind. Bussiness
   - 点击菜单”File->Add->New Project”
   - 创建名为 Bussiness 的 Class Library 项目
   - 在解决方面窗口里面选择 Bussiness 项目右键点击选择属性，打开项目属性设置窗口。
   - 修改默认名称空间名和默认程序集名修改为 Legendigital.Northwind.Bussiness。按 Ctrl+S 进行保存。
   - Bussiness 项目添加对 Common 项目以及 Entity 项目 Data 项目的引用。



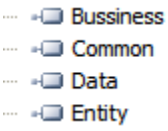
   - 确保以下第三方 Dll 被 Bussiness 项目引用。



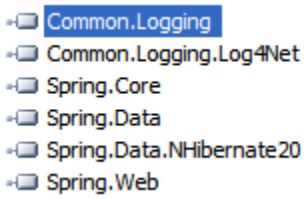
   - 删除自动生成的 class1.cs
   - 在项目里面创建 ServiceProxys-> Tables, Wrappers 目录



6. 创建 Web 表现层项目 Legendigital. Northwind. Web
    - 点击菜单”File->Add->New Project”
    - 创建名为 Web 的 Web Application 项目
    - 在解决方面窗口里面选择 Bussiness 项目右键点击选择属性，打开项目属性设置窗口。
    - 修改默认名称空间名和默认程序集名修改为 Legendigital. Northwind. Bussiness。按 Ctrl+S 进行保存。
    - Bussiness 项目添加对 Common 项目以及 Entity 项目还有 Data 项目 Bussiness 项目的引用。

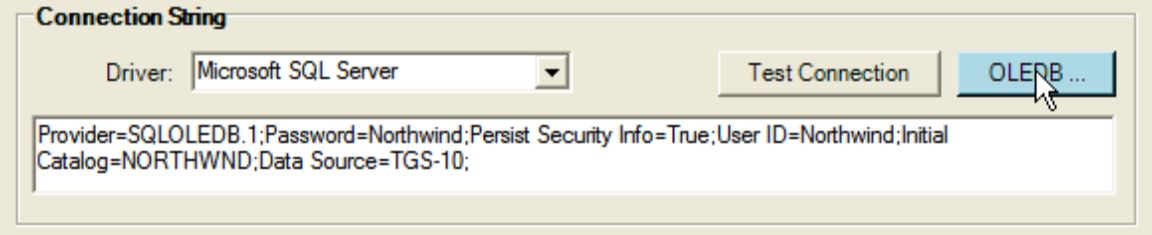
    - 确保以下第三方 Dll 被 Web 项目引用。

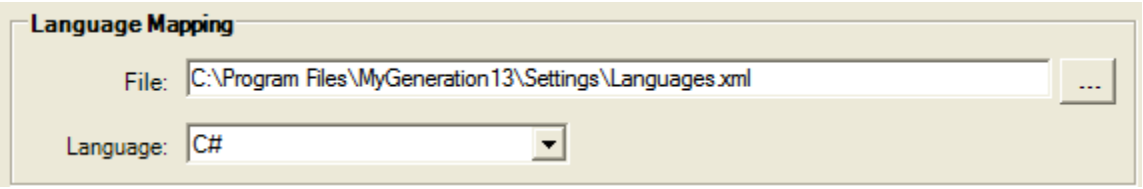
    - 删除自动生成的 Default.aspx
    - 将 Web 项目设置为启动项目

2.1.2 配置代码生成工具并生成代码

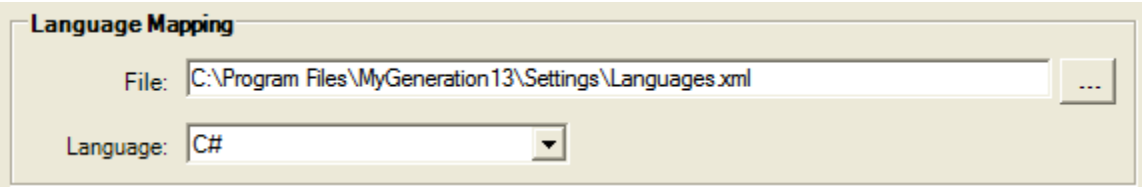
1. 安装 MyGeneration
    - 运行 mygeneration_1309_20081006.exe 安装 MyGeneration。
2. 将 Legendigital.Code.MyGenAddin 组件拷贝到 MyGeneration 安装目录下面
    - Legendigital.Code.MyGenAddin.dll 拷贝到 MyGeneration 安装目录(如：C:\Program Files\MyGeneration13) 下面
3. 配置 MyGeneration
    - 运行 MyGeneration，点击菜单 File->Default Settings
    - 设置数据库连接，Drivers 选择 Microsoft Sql Server，点击 OleDb 设置数据库连接，链接到本地的 Northwind 数据库。



- Lanauage mapping 里面设置 Lanauage 为 C#

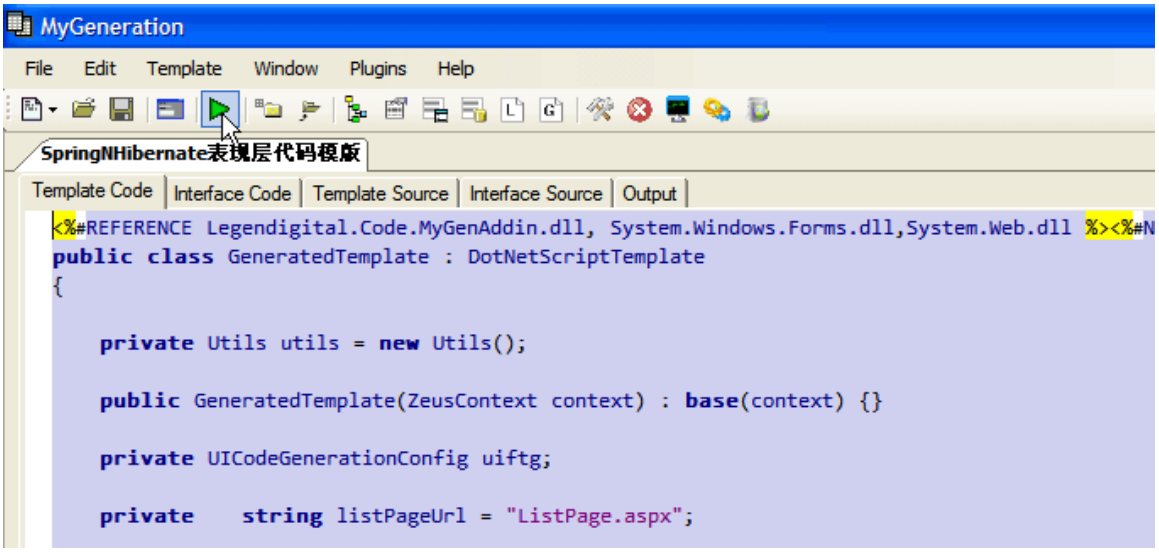


- 点击 Templates 选项卡设置 Globalization 里面的 codepage override 为 UTF-8



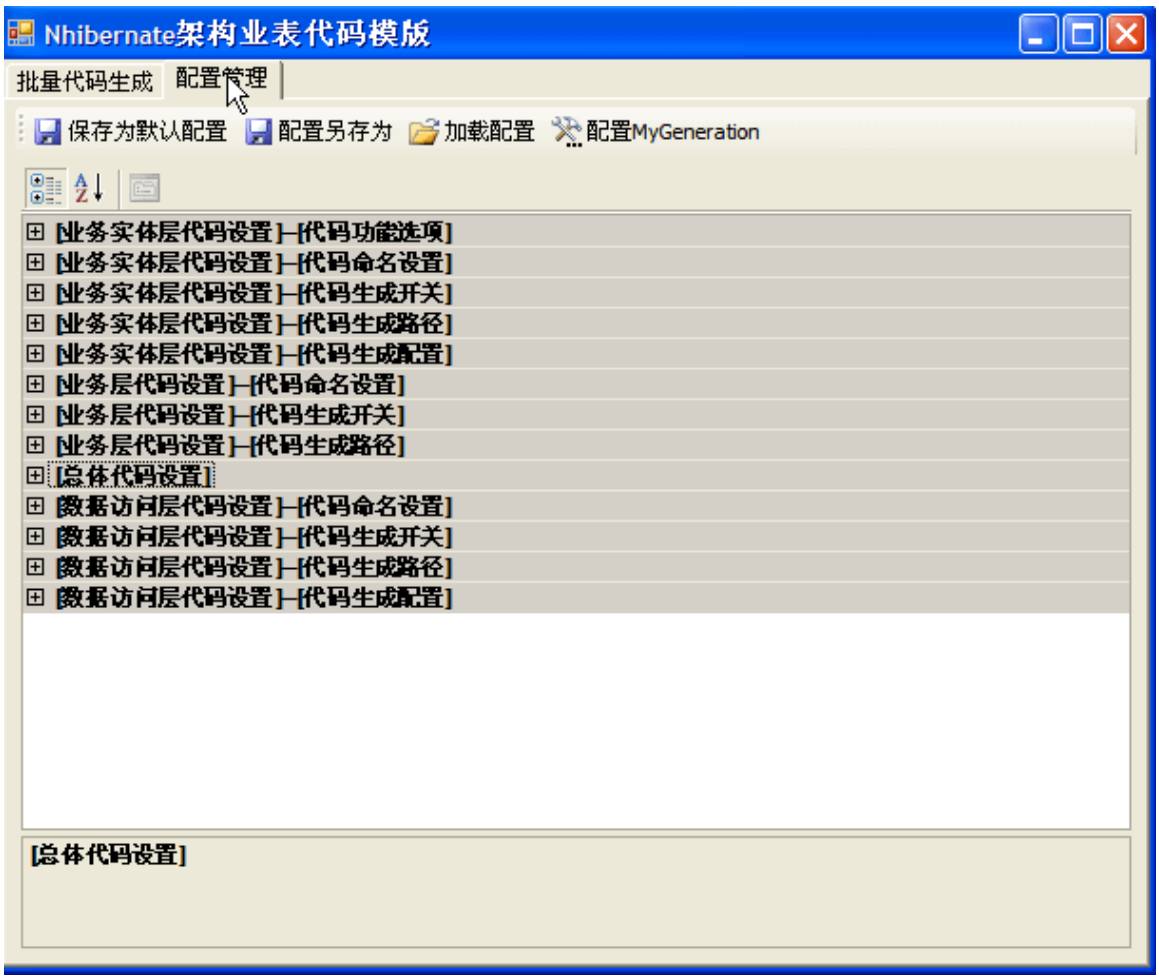
4. 运行代码模板并配置
   - 点击菜单 File->Open 打开代码生成模板文件“SpringNHibernate 开发架构表代码模板.zeus”

   - 点击运行按钮，运行模板。



- 弹出代码生成窗口

- 点击配置管理选项卡，对代码生成进行设置

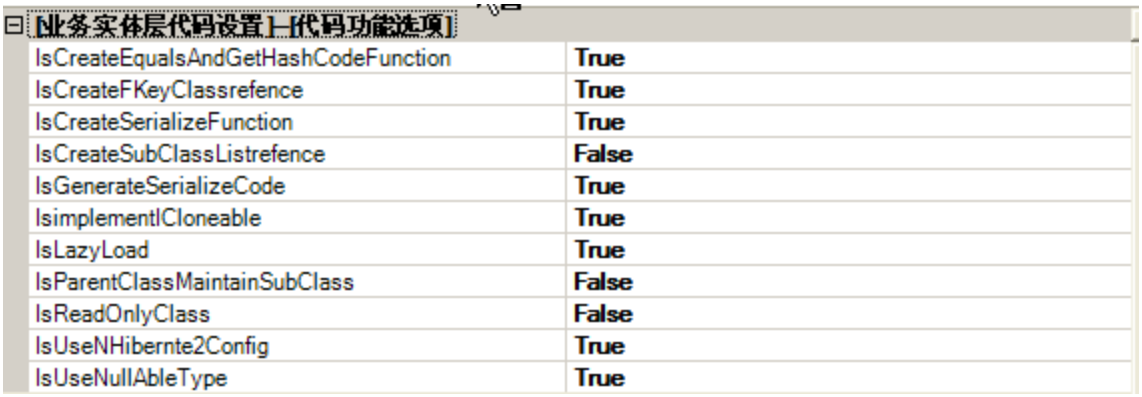


- 总体代码设置

[总体代码设置]	
DefaultDatabaseName	
IsEnbaleBussniessCode	**True**
IsEnbaleDataCode	**True**
IsEnbaleEntityCode	**True**
SelectObjectNames	

  - DefaultDatabaseName 不设置
  - IsEnbaleBussniessCode 设置为 **True**
  - IsEnbaleDataCode 设置为 **True**
  - IsEnbaleEntityCode 设置为 **True**
  - SelectObjectNames 不设置

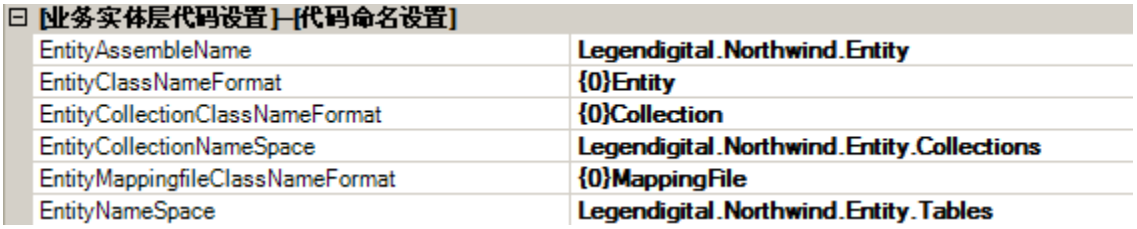
- 业务实体层代码设置-代码功能选项设置



[业务实体层代码设置]-[代码功能选项]	
IsCreateEqualsAndGetHashCodeFunction	True
IsCreateFKeyClassrefence	True
IsCreateSerializeFunction	True
IsCreateSubClassListrefence	False
IsGenerateSerializeCode	True
IsimplementICloneable	True
IsLazyLoad	True
IsParentClassMaintainSubClass	False
IsReadOnlyClass	False
IsUseNHibernte2Config	True
IsUseNullAbleType	True

- IsCreateEqualsAndGetHashCodeFunction 设置为 **True**
- IsCreateFKeyClassrefence 设置为 **True**
- IsCreateSerializeFunction 设置为 **True**
- IsCreateSubClassListrefence 设置为 **False**
- IsimplementICloneable 设置为 **True**
- IsGenerateSerializeCode 设置为 **True**
- IsLazyLoad 设置为 **True**
- IsParentClassMaintainSubClass 设置为 **False**
- IsReadOnlyClass 设置为 **False**
- IsUseNHibernte2Config 设置为 **True**
- IsUseNullAbleType 设置为 **True**

- 业务实体层代码设置-代码命名设置



[业务实体层代码设置]-[代码命名设置]	
EntityAssembleName	Legendigital.Northwind.Entity
EntityClassNameFormat	{0}Entity
EntityCollectionClassNameFormat	{0}Collection
EntityCollectionNameSpace	Legendigital.Northwind.Entity.Collections
EntityMappingfileClassNameFormat	{0}MappingFile
EntityNameSpace	Legendigital.Northwind.Entity.Tables

- EntityAssembleName 设置为 **Legendigital.Northwind.Entity**
- EntityClassNameFormat 设置为 **{0}Entity**
- EntityCollectionClassNameFormat 设置为 **{0}Collection**
- EntityCollectionNameSpace 设置为 **Legendigital.Northwind.Entity. Collections**
- EntityMappingfileClassNameFormat 设置为**{0}MappingFile**
- EntityNameSpace 设置为 **Legendigital.Northwind.Entity. Tables**

- 业务实体层代码设置-代码生成开关

[业务实体层代码设置]-[代码生成开关]	
IsCreateCollection	**False**
IsCreateEntityClassFile	**True**
IsCreateMappingFile	**True**

  - IsCreateCollection 设置为 **False**
  - IsCreateEntityClassFile 设置为 **True**
  - IsCreateMappingFile 设置为 **True**

- 业务实体层代码设置-代码生成路径

  - EntityCollectionNameSpace 设置为 **Legendigital.Northwind.Entity. Collections**
  - EntityCollectionNameSpace 设置为 **Legendigital.Northwind.Entity. Collections**
  - EntityCollectionNameSpace 设置为 **Legendigital.Northwind.Entity. Collections**

- 业务实体层代码设置-代码生成配置

[业务实体层代码设置]-[代码生成配置]	
DbPrefix	_
IsUseSequenceKey	**False**
MemberPrefix	_
SequenceIDNameFormat	
SpecifiedPrimaryKeyIndex	**0**

- 数据访问层代码设置--代码命名设置

[数据访问层代码设置]-[代码命名设置]	
DataObjectAssembleName	**Legendigital.Demo.Data**
DataObjectClassNameFormat	**{0}DataObject**
DataObjectContainerIoCClassName	**DataObjectContainers**
DataObjectContainerIoCClassNameSpace	**Legendigital.Demo.Data.Tables.Container**
DataObjectIocXmlFileName	**DataObjectContainersFile**
DataObjectNameSpace	**Legendigital.Demo.Data.Tables**

- 数据访问层代码设置--代码生成开关

[数据访问层代码设置]-[代码生成开关]	
IsCreateDataObjectClassFile	**True**
IsCreateDataObjectContainerIoCClass	**True**
IsCreateDataObjectIocXmlFileName	**True**

- 数据访问层代码设置--代码生成路径

[数据访问层代码设置]-[代码生成路径]	
DataObjectContainerIocClassFilePath	**D:\My Project\Legendigital.Framework.Demo\Leg**
DataObjectIocXmlFilePath	**D:\My Project\Legendigital.Framework.Demo\Leg**
SelectDataObjectClassFilePath	**D:\My Project\Legendigital.Framework.Demo\Leg**

- 数据访问层代码设置--代码生成配置

[数据访问层代码设置]-[代码生成配置]	
ImportNameSpace	**using System;using System.Collections.Generic;us**
RootDataObjectName	**BaseNHibernateDataObject**
RootDataObjectNameSpace	**Legendigital.Framework.Common.Data.NHibernat**

- 业务层代码设置--代码命名设置

[业务层代码设置]--[代码命名设置]	
BussinessAssembleName	Legendigital.Demo.Bussiness
GenerateBussinessEncapsulationClassNameFormat	{0}Wrapper
GenerateBussinessEncapsulationClassNameSpace	Legendigital.Demo.Bussiness.Wrappers
GenerateServiceProxyContainerClassName	ServiceProxyContainer
GenerateServiceProxyContainerClassNameSpace	Legendigital.Demo.Bussiness.ServiceProxys.Tabl
GenerateServiceProxyContainerXMLFileName	ServiceProxyContainersFile
ImportServiceProxyNameSpace	using System;using System.Collections.Generic;us
RootServiceProxyClassName	BaseSpringNHibernateEntityServiceProxy
RootServiceProxyClassNameSpace	Legendigital.Framework.Common.Bussiness.NHib
ServiceProxyClassNameFormat	{0}ServiceProxy
ServiceProxyInterfaceNameFormat	I{0}ServiceProxy
ServiceProxyNameSpace	Legendigital.Demo.Bussiness.ServiceProxys.Tabl

- 业务层层代码设置--代码生成开关

[业务层代码设置]--[代码生成开关]	
IsCreateServiceProxyClassFile	False
IsGenerateBussinessEncapsulationClass	True
IsGenerateServiceProxyContainerClass	False

- 业务层层代码设置--代码生成路径

[业务层代码设置]--[代码生成路径]	
SelectGenerateBussinessEncapsulationClassFilePat	D:\My Project\Legendigital.Framework.Demo\Leg
SelectGenerateServiceProxyContainerClassFilePath	D:\My Project\Legendigital.Framework.Demo\Leg
SelectGenerateServiceProxyContainerClassXmlFileF	D:\My Project\Legendigital.Framework.Demo\Leg
SelectServiceProxyClassFilePath	D:\My Project\Legendigital.Framework.Demo\Leg

5. 自动生成业务层实体层数据层代码

选中需要生成代码的表点击代码生成按钮，生成所有的代码。

2.1.3 配置系统设置并运行示例

1. 配置 Web.Config
2. 配置数据库
3. 配置日志
4. 运行系统

2.1.4 完成一个最简单的 CURD（增删改查）功能

1. 编写列表页面
2. 编写添加页面
3. 编写修改页面
4. 运行系统

3 代码生成工具的使用

3.1 MyGeneration 简介

3.2 NHibernate 架构表代码模版介绍

3.2.1 使用简介

3.2.2 配置参数详解

3.3 NHibernate 架构表现层代码模版

3.3.1 使用简介

3.3.2 配置参数详解