DDD领域驱动设计初探（7）：Web层的搭建 - 文章 - 伯乐在线

前言：好久没更新博客了，每天被该死的业务缠身，今天正好一个模块完成了，继续来完善我们的代码。之前的六篇完成了领域层、应用层、以及基础结构层的部分代码，这篇打算搭建下UI层的代码。

一、UI层介绍

在DDD里面，UI层的设计也分为BS和CS，本篇还是以Web为例来说明。我们的Web采用的是MVC+bootstrap的架构。Table组件使用的是bootstrap table，之所以用它是因为它的API比较全，并且博主觉得它的风格适用于各种类型的设备，无论是PC端还是手机端都都能很好的兼容各种浏览器。

这里还是贴出bootstrap API的相关地址。

Bootstrap中文网：http://www.bootcss.com/

Bootstrap Table Demo：http://issues.wenzhixin.net.cn/bootstrap-table/index.html

Bootstrap Table API：http://bootstrap-table.wenzhixin.net.cn/zh-cn/documentation/

Bootstrap Table源码：https://github.com/wenzhixin/bootstrap-table

Bootstrap DataPicker：http://www.bootcss.com/p/bootstrap-datetimepicker/

二、代码示例

上篇完成了WCF的设计代码，但是具体的业务逻辑的代码还没有，我们先来实现具体业务的CURD代码。

1、WCF代码

1.1 WCF服务业务接口代码

C#

```
        ///
        /// 权限管理模块接口契约
        ///
        [ServiceContract]
        [ServiceInterface]
        public interface IPowerManageWCFService
```

```
        {
                #region 用户管理
                [OperationContract]
                List GetUsers(ExpressionNode expressionNode);
                [OperationContract]
                DTO_TB_USERS AddUser(DTO_TB_USERS oUser);
                [OperationContract]
                bool DeleteUser(DTO_TB_USERS oUser);
                [OperationContract]
                bool DeleteUserByLamada(ExpressionNode expressionNode);
                [OperationContract]
                bool UpdateUser(DTO_TB_USERS oUser);
                #endregion
                #region 部门管理
                [OperationContract]
                List GetDepartments(ExpressionNode expressionNode);
                [OperationContract]
                DTO_TB_DEPARTMENT AddDepartment(DTO_TB_DEPARTMENT oDept);
                [OperationContract]
                bool DeleteDepartment(DTO_TB_DEPARTMENT oDept);
                [OperationContract]
                bool DeleteDeptByLamada(ExpressionNode expressionNode);
                [OperationContract]
                bool UpdateDepartment(DTO_TB_DEPARTMENT oDept);
                #endregion
                #region 角色管理
                [OperationContract]
                List GetRoles(ExpressionNode expressionNode);
                [OperationContract]
                DTO_TB_ROLE AddRole(DTO_TB_ROLE oRole);
                #endregion
                #region 菜单管理
                [OperationContract]
                List GetMenus(ExpressionNode expressionNode);
                [OperationContract]
                DTO_TB_MENU AddMenu(DTO_TB_MENU oMenu);
                #endregion
        }
```

C#

```
[ServiceClass]
        public class PowerManageWCFService :BaseService, IPowerManageWCFService
        {
```

```
#region Fields
[Import]
private IUserRepository userRepository { get; set; }
[Import]
private IDepartmentRepository departmentRepository { get; set; }
[Import]
private IRoleRepository roleRepository { get; set; }
[Import]
private IMenuRepository menuRepository { get; set; }
#endregion
#region Constust
public PowerManageWCFService()
{
}
#endregion
#region WCF服务接口实现
#region 用户管理
//这里参数为什么不直接用Expression>这种类型，是因为Expression不支持序列化，无法用于WCF数据的传递
public List GetUsers(ExpressionNode expressionNode)
{
        Expressionbool>> selector = null;
        if (expressionNode != null)
        {
                selector = expressionNode.ToExpressionbool>>();
        }
        var lstRes = base.GetDtoByLamada(userRepository, selector);
        return lstRes;
}
public DTO_TB_USERS AddUser(DTO_TB_USERS oUser)
{
        return base.AddDto(userRepository, oUser);
}
public bool DeleteUser(DTO_TB_USERS oUser)
{
        var bRes = false;
        try
        {
                base.DeleteDto(userRepository, oUser);
                bRes = true;
        }
        catch
```

```
        {
        }
        return bRes;
}
public bool DeleteUserByLamada(ExpressionNode expressionNode)
{
        Expressionbool>> selector = null;
        if (expressionNode != null)
        {
                selector = expressionNode.ToExpressionbool>>();
        }
        var bRes = false;
        try
        {
                base.DeleteDto(userRepository, selector);
                bRes = true;
        }
        catch
        {
        }
        return bRes;
}
public bool UpdateUser(DTO_TB_USERS oUser)
{
        var bRes = false;
        try
        {
                base.UpdateDto(userRepository, oUser);
                bRes = true;
        }
        catch
        {
        }
        return bRes;
}
#endregion
#region 部门管理
public List GetDepartments(ExpressionNode expressionNode)
{
        Expressionbool>> selector = null;
        if (expressionNode != null)
        {
```

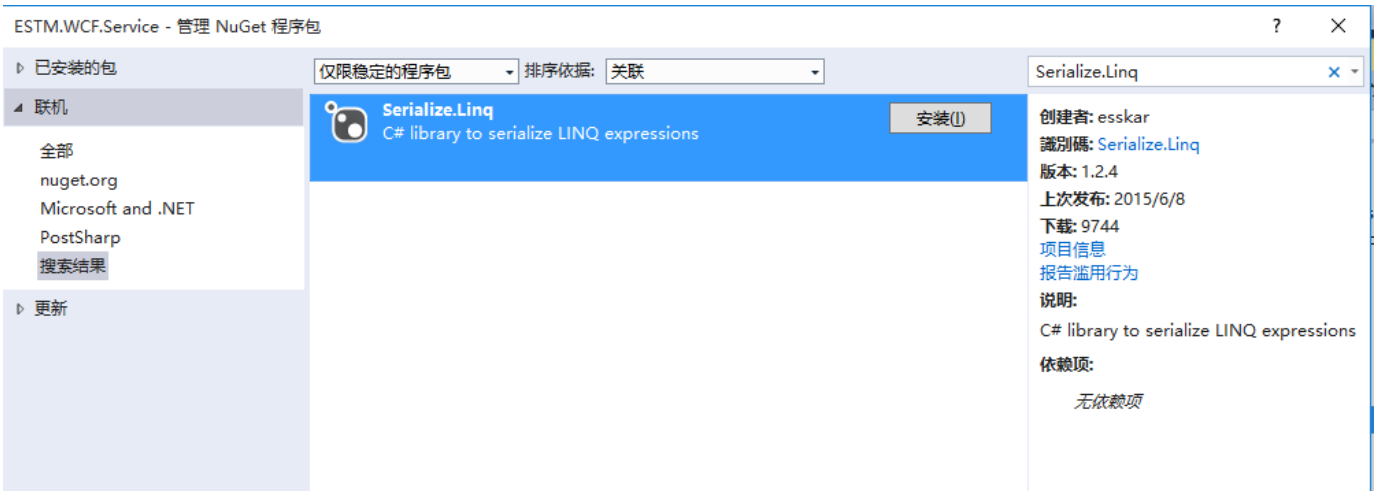
```
                selector = expressionNode.ToExpressionbool>>();
        }
        return base.GetDtoByLamada(departmentRepository, selector);
}
public DTO_TB_DEPARTMENT AddDepartment(DTO_TB_DEPARTMENT oDept)
{
        return base.AddDto(departmentRepository, oDept);
}
public bool DeleteDepartment(DTO_TB_DEPARTMENT oDept)
{
        var bRes = false;
        try
        {
                base.DeleteDto(departmentRepository, oDept);
                bRes = true;
        }
        catch
        {
        }
        return bRes;
}
public bool DeleteDeptByLamada(ExpressionNode expressionNode)
{
        Expressionbool>> selector = null;
        if (expressionNode != null)
        {
                selector = expressionNode.ToExpressionbool>>();
        }
        var bRes = false;
        try
        {
                base.DeleteDto(departmentRepository, selector);
                bRes = true;
        }
        catch
        {
        }
        return bRes;
}
public bool UpdateDepartment(DTO_TB_DEPARTMENT oDept)
{
        var bRes = false;
```

```
            try
            {
                    base.UpdateDto(departmentRepository, oDept);
                    bRes = true;
            }
            catch
            {
            }
            return bRes;
    }
    #endregion
    #region 角色管理
    public List GetRoles(ExpressionNode expressionNode)
    {
            Expressionbool>> selector = null;
            if (expressionNode != null)
            {
                    selector = expressionNode.ToExpressionbool>>();
            }
            return base.GetDtoByLamada(roleRepository, selector);
    }
    public DTO_TB_ROLE AddRole(DTO_TB_ROLE oRole)
    {
            return base.AddDto(roleRepository, oRole);
    }
    #endregion
    #region 菜单管理
    public List GetMenus(ExpressionNode expressionNode)
    {
            Expressionbool>> selector = null;
            if (expressionNode != null)
            {
                    selector = expressionNode.ToExpressionbool>>();
            }
            return base.GetDtoByLamada(menuRepository, selector);
    }
    public DTO_TB_MENU AddMenu(DTO_TB_MENU oMenu)
    {
            return base.AddDto(menuRepository, oMenu);
    }
    #endregion
    #endregion
```

```
        }
```

这里要说明一点，在通过lamada表达式查询的方法里面为什么不直接用Expression>这种类型，而要使用ExpressionNode这种类型的变量呢？

这是因为Expression不支持序列化，无法用于WCF数据的传递。ExpressionNode这个对象的使用需要添加Serialize.Linq这个dll的引用，还好有我们神奇的NuGet，让我们再也不用去网上找一大堆的dll了。



我们公用的增删改查封装到了BaseService这个父类里面。

1.3 BaseService代码

```
public class BaseService
        {
                #region Fields
                private bool bInitAutoMapper = false;
                #endregion
                #region Construct
                public BaseService()
                {
                        //注册MEF
                        Regisgter.regisgter().ComposeParts(this);
                }
                #endregion
                #region 查询
                ///
                /// 通用单表查询方法
                ///
                /// DTOmodel
                /// 领域模型
                /// 需要传过来的仓储接口对象
                /// 前端传过来的lamada表达式
                ///
                public List GetDtoByLamada(IRepository oRepository, Expressionbool>>
```

```
selector = null)
                        where DomainModel : AggregateRoot
                        where DtoModel : Dto_BaseModel
                {
                        InitAutoMapper();
                        if (selector == null)
                        {
                                var lstDomainModel = oRepository.Entities.ToList();
                                return Mapper.Map, List>(lstDomainModel);
                        }
                        //得到从Web传过来和DTOModel相关的lamaba表达式的委托
                        Funcbool> match = selector.Compile();
                        //创建映射Expression的委托
                        Func mapper =
AutoMapper.QueryableExtensions.Extensions.CreateMapExpression(Mapper.Engine).Compile();
                        //得到领域Model相关的lamada
                        Expressionbool>> lamada = ef_t => match(mapper(ef_t));
                        List list = oRepository.Find(lamada).ToList();
                        return Mapper.Map, List>(list);
                }
                #endregion
                #region 新增
                public DtoModel AddDto(IRepository oRepository, DtoModel oDtoModel)
                        where DomainModel : AggregateRoot
                        where DtoModel : Dto_BaseModel
                {
                        InitAutoMapper();
                        var oDomain = Mapper.Map(oDtoModel);
                        oRepository.Insert(oDomain);
                        return Mapper.Map(oDomain);
                }
                #endregion
                #region 删除
                public int DeleteDto(IRepository oRepository, DtoModel oDtoModel)
                        where DomainModel : AggregateRoot
                        where DtoModel : Dto_BaseModel
                {
                        InitAutoMapper();
                        var oDomain = Mapper.Map(oDtoModel);
                        return oRepository.Delete(oDomain);
                }
                public int DeleteDto(IRepository oRepository, Expressionbool>> selector =
```

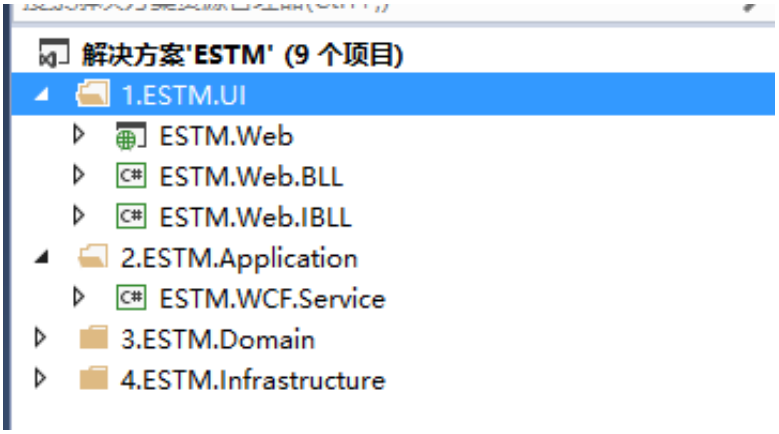
```
null)
                        where DomainModel : AggregateRoot
                        where DtoModel : Dto_BaseModel
                {
                        InitAutoMapper();
                        if (selector == null)
                        {
                                return 0;
                        }
                        //得到从Web传过来和DTOModel相关的lamaba表达式的委托
                        Funcbool> match = selector.Compile();
                        //创建映射Expression的委托
                        Func mapper =
AutoMapper.QueryableExtensions.Extensions.CreateMapExpression(Mapper.Engine).Compile();
                        //得到领域Model相关的lamada
                        Expressionbool>> lamada = ef_t => match(mapper(ef_t));
                        return oRepository.Delete(lamada);
                }
                #endregion
                #region 更新
                public void UpdateDto(IRepository oRepository, DtoModel oDtoModel)
                        where DomainModel : AggregateRoot
                        where DtoModel : Dto_BaseModel
                {
                        InitAutoMapper();
                        var oDomain = Mapper.Map(oDtoModel);
                        oRepository.Update(oDomain);
                }
                #endregion
                #region Private
                private void InitAutoMapper()
                {
                        var oType = Mapper.FindTypeMapFor();
                        if (oType==null)
                        {
                                Mapper.CreateMap();
                                Mapper.CreateMap();
                        }
                }
                #endregion
        }
```

这个父类主要做了两件事：一是MEF的初始化；二是通用增删改查的实现。所有dto对象和领域model的映

射都在这里统一管理。

2、UI层代码

UI层里面，为了更好分离代码，我们引入了接口编程的机制，引入了ESTM.Web.IBLL和ESTM.Web.BLL两个项目，如图：



为什么要有这么一个接口层？之前 MEF实现设计上的“松耦合”（终结篇：面向接口编程）这篇已经做过介绍，对面向接口编程不了解的朋友可以看看。

2.1 ESTM.Web.IBLL代码

这个dll主要定义接口规则。

C#

```
public interface IPowerManager
        {
                List GetUsers(Expressionbool>> selector = null);
                DTO_TB_USERS AddUser(DTO_TB_USERS oUser);
                bool DeleteUser(DTO_TB_USERS oUser);
                bool UpdateUser(DTO_TB_USERS oUser);
                bool DeleteUser(Expressionbool>> selector = null);
                List GetDepartments(Expressionbool>> selector = null);
                DTO_TB_DEPARTMENT AddDepartment(DTO_TB_DEPARTMENT oDept);
                bool DeleteDepartment(DTO_TB_DEPARTMENT oDept);
                bool DeleteDepartment(Expressionbool>> selector = null);
                bool UpdateDepartment(DTO_TB_DEPARTMENT oDept);
                List GetRoles(Expressionbool>> selector = null);
                List GetMenus(Expressionbool>> selector = null);
        }
```

C#

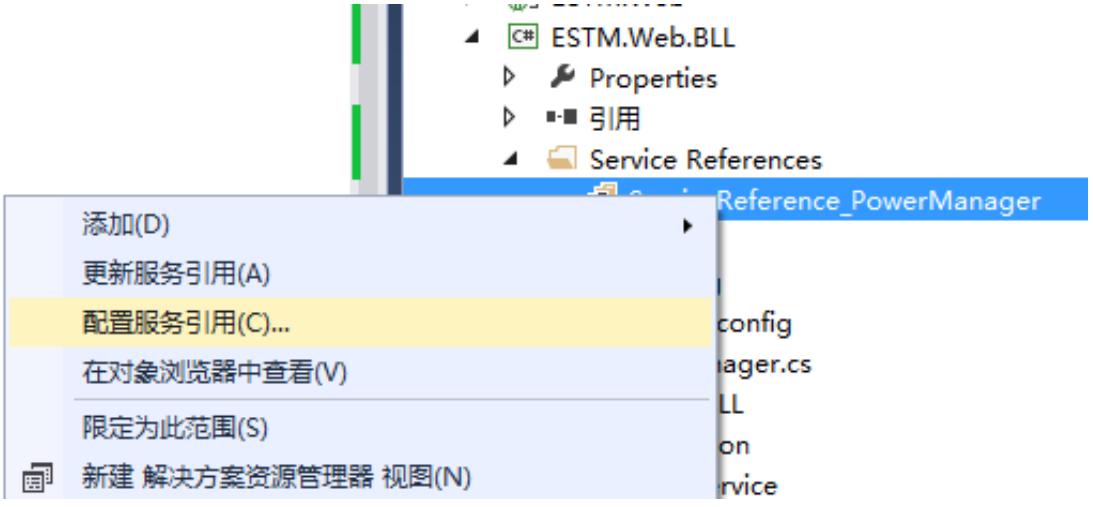
```
[Export(typeof(IPowerManager))]
        public class PowerManager : IPowerManager
        {
                #region Fields
```

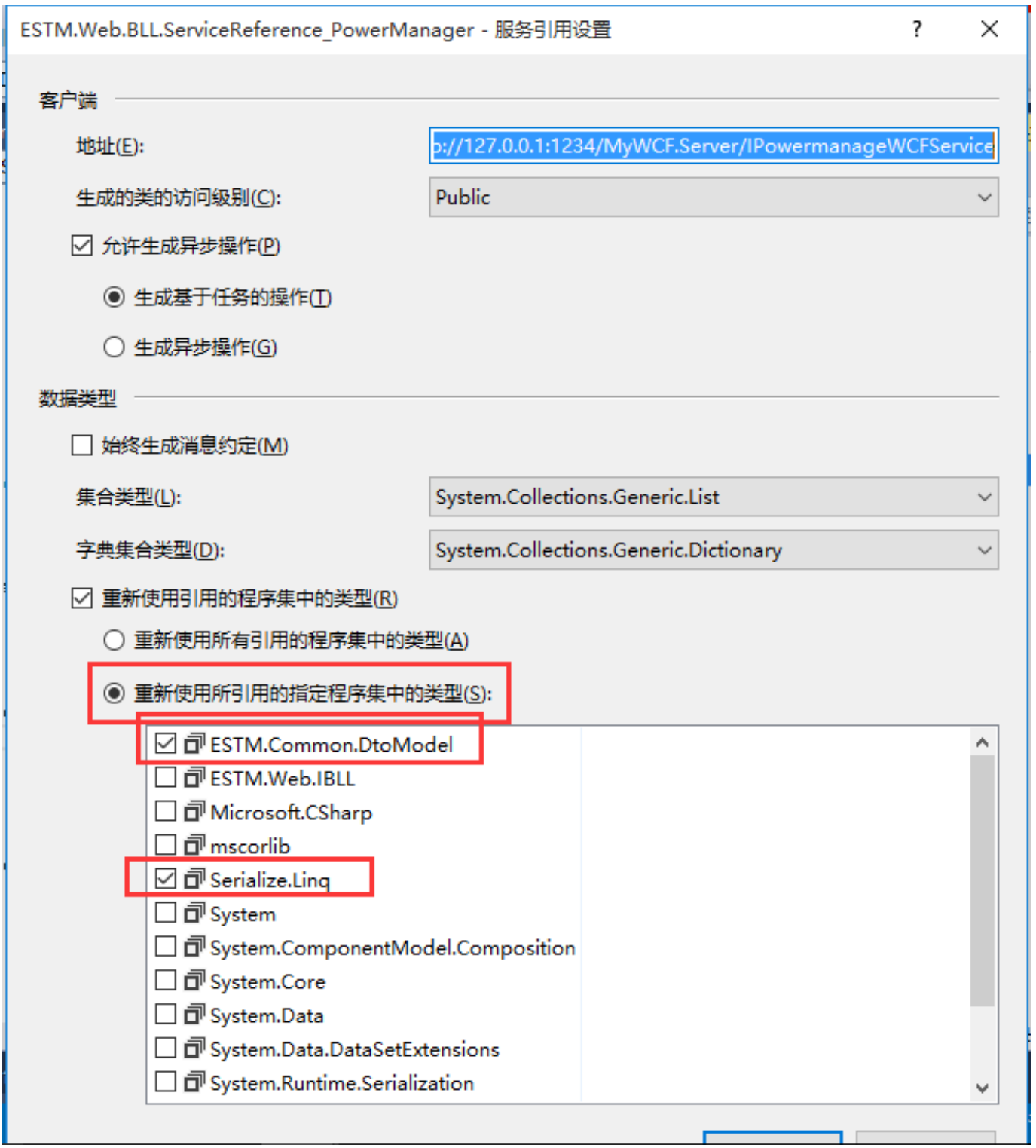
```
        //创建WCF服务连接对象
                private ServiceReference_PowerManager.PowerManageWCFServiceClient oService
= CreatePowerManagerService.GetInstance();
                #endregion
                #region 接口实现
                public List GetUsers(Expressionbool>> selector = null)
                {
                        return oService.GetUsers(GetExpressionNode(selector));
                }
                public List GetDepartments(Expressionbool>> selector = null)
                {
                        return oService.GetDepartments(GetExpressionNode(selector));
                }
                public List GetRoles(Expressionbool>> selector = null)
                {
                        return oService.GetRoles(GetExpressionNode(selector));
                }
                public List GetMenus(Expressionbool>> selector = null)
                {
                        return oService.GetMenus(GetExpressionNode(selector));
                }
                #endregion
                #region Privates
                //将lamada表达式转换为可用于WCF传递的ExpressionNode类型
                private ExpressionNode GetExpressionNode(Expressionbool>> selector)
                {
                        if (selector == null)
                        {
                                return null;
                        }
                        ExpressionConverter expressionConverter = new
ExpressionConverter();
                        ExpressionNode expressionNode =
expressionConverter.Convert(selector);
                        return expressionNode;
                }
                #endregion
                public DTO_TB_USERS AddUser(DTO_TB_USERS oUser)
                {
                        return oService.AddUser(oUser);
                }
                public bool DeleteUser(DTO_TB_USERS oUser)
```

```
                {
                        return oService.DeleteUser(oUser);
                }
                public bool DeleteUser(Expressionbool>> selector = null)
                {
                        if (selector == null)
                        {
                                return false;
                        }
                        ExpressionConverter expressionConverter = new
ExpressionConverter();
                        ExpressionNode expressionNode =
expressionConverter.Convert(selector);
                        return oService.DeleteUserByLamada(expressionNode);
                }
                public bool UpdateUser(DTO_TB_USERS oUser)
                {
                        return oService.UpdateUser(oUser);
                }
                public DTO_TB_DEPARTMENT AddDepartment(DTO_TB_DEPARTMENT oDept)
                {
                        return oService.AddDepartment(oDept);
                }
                public bool DeleteDepartment(DTO_TB_DEPARTMENT oDept)
                {
                        return oService.DeleteDepartment(oDept);
                }
                public bool UpdateDepartment(DTO_TB_DEPARTMENT oDept)
                {
                        return oService.UpdateDepartment(oDept);
                }
                public bool DeleteDepartment(Expressionbool>> selector = null)
                {
                        if (selector == null)
                        {
                                return false;
                        }
                        ExpressionConverter expressionConverter = new
ExpressionConverter();
                        ExpressionNode expressionNode =
expressionConverter.Convert(selector);
                        return oService.DeleteDeptByLamada(expressionNode);
```

```
                }
        }
        public class CreatePowerManagerService
        {
                private static ServiceReference_PowerManager.PowerManageWCFServiceClient
oPowerManagerClient = null;
                private static object obj = new object();
                public static ServiceReference_PowerManager.PowerManageWCFServiceClient
GetInstance()
                {
                        lock (obj)
                        {
                                if (oPowerManagerClient == null)
                                {
                                        oPowerManagerClient = new
ServiceReference_PowerManager.PowerManageWCFServiceClient();
                                }
                        }
                        return oPowerManagerClient;
                }
        }
```

由于是采用的添加服务引用的方式引用的WCF服务，所以在这一层需要添加WCF服务的引用。在实现这部分代码的时候博主遇到过一个问题，在此和朋友们分享一下。由于在WCF服务的设计里面用到了DTO对象，而在ESTM.Web.BLL这个项目里面也要用到DTO，但是添加WCF服务引用的时候默认的是WCF服务里面的DTO，而不是ESTM.Common.DtoModel这个项目的DTO对象，这样就有问题了，每次如果我们需要改动下dto的内容，那么我们就需要更新下服务引用。还好，微软给我们选择的机制，我们来看图





这样就能解决上面的问题了。

2.3 ESTM.Web代码

按照面向接口的机制，ESTM.Web项目是不需要添加ESTM.Web.BLL这个实现层项目引用的，通过MEF动态导入ESTM.Web.BLL里面的对象。我们来看代码：

C#

```
public class PowerManagerController : BaseController
        {
                [Import]
                private IPowerManager PowerManager { set; get; }
```

```
        #region Views
        // GET: PowerManager
        public ActionResult User()
        {
                return View();
        }
        public ActionResult Role()
        {
                return View();
        }
        public ActionResult Menu()
        {
                return View();
        }
        public ActionResult Department()
        {
                return View();
        }
        #endregion
        #region 部门管理
        public JsonResult GetDepartments(int limit, int offset, string
departmentname, string statu)
        {
                //得到lamada表达式
                var oLamadaExtention = new LamadaExtention();
                if (!string.IsNullOrEmpty(departmentname))
                {
                        oLamadaExtention.GetExpression("DEPARTMENT_NAME",
departmentname, ExpressionType.Contains);
                }
                if (!string.IsNullOrEmpty(statu))
                {
                        oLamadaExtention.GetExpression("STATUS", statu,
ExpressionType.Contains);
                }
                var lamada = oLamadaExtention.GetLambda();
                var lstRes = PowerManager.GetDepartments(lamada);
                return Json(new { rows = lstRes.Skip(offset).Take(limit).ToList(),
total = lstRes.Count }, JsonRequestBehavior.AllowGet);
        }
        public object GetDepartmentEdit(string strPostData)
        {
```

```
                        var oDepartment =
Newtonsoft.Json.JsonConvert.DeserializeObject(strPostData);
                        if (string.IsNullOrEmpty(oDepartment.DEPARTMENT_ID))
                        {
                                oDepartment.DEPARTMENT_ID = Guid.NewGuid().ToString();
                                oDepartment = PowerManager.AddDepartment(oDepartment);
                        }
                        else
                        {
                                PowerManager.UpdateDepartment(oDepartment);
                        }
                        return oDepartment;
                }
                public object DeleteDept(string strID)
                {
                        PowerManager.DeleteDepartment(x=>x.DEPARTMENT_ID == strID);
                        return new object();
                }
                #endregion
                #region 菜单管理
                public JsonResult GetMenus(int limit, int offset, string menuname, string
menuurl)
                {
                        var oLamadaExtention = new LamadaExtention();
                        if (!string.IsNullOrEmpty(menuname))
                        {
                                oLamadaExtention.GetExpression("MENU_NAME", menuname,
ExpressionType.Contains);
                        }
                        if (!string.IsNullOrEmpty(menuurl))
                        {
                                oLamadaExtention.GetExpression("MENU_URL", menuurl,
ExpressionType.Contains);
                        }
                        var lamada = oLamadaExtention.GetLambda();
                        var lstRes = PowerManager.GetMenus(lamada).ToList();
                        return Json(new { rows = lstRes.Skip(offset).Take(limit).ToList(),
total = lstRes.Count }, JsonRequestBehavior.AllowGet);
                }
                public object GetMenuEdit(string strPostData)
                {
                        var oMenu =
```

```
Newtonsoft.Json.JsonConvert.DeserializeObject(strPostData);
                if (string.IsNullOrEmpty(oMenu.MENU_ID))
                {
                        //oMenu = MenuManager.Add(oMenu);
                }
                else
                {
                        //MenuManager.Update(oMenu);
                }
                return oMenu;
        }
        public object DeleteMenu(string strID)
        {
                //MenuManager.Delete(strID);
                return new object();
        }
        public object GetParentMenu()
        {
                var lstMenu = PowerManager.GetMenus(x => x.MENU_LEVEL == "1");
                //var lstRes = RoleManager.Find().ToList();
                //var oRes = new PageRowData();
                //oRes.rows = lstRes.Skip(offset).Take(limit).ToList();
                //oRes.total = lstRes.Count;
                return lstMenu; ;
        }
        public object GetChildrenMenu(string strParentID)
        {
                var lstMenu = PowerManager.GetMenus(x => x.MENU_LEVEL == "2" &
x.PARENT_ID == strParentID).ToList();
                //var lstRes = RoleManager.Find().ToList();
                //var oRes = new PageRowData();
                //oRes.rows = lstRes.Skip(offset).Take(limit).ToList();
                //oRes.total = lstRes.Count;
                return lstMenu; ;
        }
        #endregion
        #region 权限管理
        public JsonResult GetRole(int limit, int offset, string rolename, string
desc)
        {
                var oLamadaExtention = new LamadaExtention();
                if (!string.IsNullOrEmpty(rolename))
```

```
                        {
                                oLamadaExtention.GetExpression("ROLE_NAME", rolename,
ExpressionType.Contains);
                        }
                        if (!string.IsNullOrEmpty(desc))
                        {
                                oLamadaExtention.GetExpression("DESCRIPTION", desc,
ExpressionType.Contains);
                        }
                        var lamada = oLamadaExtention.GetLambda();
                        var lstRes = PowerManager.GetRoles(lamada).ToList();
                        return Json(new { rows = lstRes.Skip(offset).Take(limit).ToList(),
total = lstRes.Count }, JsonRequestBehavior.AllowGet);
                }
                #endregion
                #region 用户管理
                public JsonResult GetUsers(int limit, int offset, string username, string
fullname)
                {
                        var oLamadaExtention = new LamadaExtention();
                        if (!string.IsNullOrEmpty(username))
                        {
                                oLamadaExtention.GetExpression("USER_NAME", username,
ExpressionType.Contains);
                        }
                        if (!string.IsNullOrEmpty(fullname))
                        {
                                oLamadaExtention.GetExpression("FULLNAME", fullname,
ExpressionType.Contains);
                        }
                        var lamada = oLamadaExtention.GetLambda();
                        var lstRes = PowerManager.GetUsers(lamada).ToList();
                        return Json(new { rows = lstRes.Skip(offset).Take(limit).ToList(),
total = lstRes.Count }, JsonRequestBehavior.AllowGet);
                }
                public object GetUserEdit(string strPostData)
                {
                        var oUser =
Newtonsoft.Json.JsonConvert.DeserializeObject(strPostData);
                        if (string.IsNullOrEmpty(oUser.USER_ID))
                        {
                                oUser.USER_ID = Guid.NewGuid().ToString();
```

```
                oUser = PowerManager.AddUser(oUser);
            }
            else
            {
                PowerManager.UpdateUser(oUser);
            }
            return oUser;
        }
        public object DeleteUser(string strID)
        {
            PowerManager.DeleteUser(x => x.USER_ID == strID);
            return new object();
        }
        #endregion
    }
```