三行统一业务平台（TG.UBP.OSharp） 开发说明

1. 开发工具：

VS2017(15.2)

Oracle 11g

github for vs2017

devart oracle pro\_9.2.187

1. 采用技术：

MVC5、EF6、AutoFac（依赖注入）、AutoMapper（实体和DTO的映射）、Web API 2、SignalR、HangFire（后台任务）、EasyUI、Redis、OAuth……

1. 采用的框架或参照的原型：

TG.UBP.OSharp的后台采用OSharp、前台参照Ymnets、功能参照力软（Learun），具体说明如下：

OSharp：OSharp是个快速开发框架，但不是一个大而全的包罗万象的框架，严格的说，OSharp中什么都没有实现。与其他大而全的框架最大的不同点，就是OSharp只做抽象封装，不做实现。依赖注入、ORM、对象映射、日志、缓存等等功能，都只定义了一套最基础最通用的抽象封装，提供了一套统一的API、约定与规则，并定义了部分执行流程，主要是让项目在一定的规范下进行开发。所有的功能实现端，都是通过现有的成熟的第三方组件来实现的，除了EntityFramework之外，所有的第三方实现都可以轻松的替换成另一种第三方实现，OSharp框架正是要起隔离作用，保证这种变更不会对业务代码造成影响，使用统一的API来进行业务实现，解除与第三方实现的耦合，保持业务代码的规范与稳定。

1. 分层设计：

借鉴了DDD的分层思路，共分四层：

**基础设施层（Infrastructure）**：当在领域层中为定义了仓储接口，应该在基础设施层中实现这些接口。这里使用EntityFramework。数据库迁移也被用于这一层。

具体规则：

1、基础设施层主要包括以下内容：

DbContext

数据库迁移

......

2、OSharp框架只实现了MSSql、MySql的数据存储，所以增加了对Oracle的支持；

3、由于Oracle官方的EF存在命名大小写和引号等问题，所以采用了Devart的Oracle组件；

4、参照OSharp框架的代码，实现了基于Devart for Oracle的一些基类；

**领域层**：领域层就是业务层，是一个项目的核心，所有业务规则都应该在领域层实现。

**实体（Entity）**：实体代表业务领域的数据和操作，在实践中，通过用来映射成数据库表。

**仓储（Repository）**：仓储用来操作数据库进行数据存取。仓储接口在领域层定义，而仓储的实现类应该写在基础设施层。

**领域服务（Domain service）**：当处理的业务规则跨越两个（及以上）实体时，应该写在领域服务方法里面。

**领域事件（Domain Event）**：在领域层某些特定情况发生时可以触发领域事件，并且在相应地方捕获并处理它们。

**工作单元（Unit of Work）**：工作单元是一种设计模式，用于维护一个由已经被修改(如增加、删除和更新等)的业务对象组成的列表。它负责协调这些业务对象的持久化工作及并发问题。

具体规则：

1、领域层主要包括以下内容：

实体：

实体映射（由于不同数据库之间存在一定的不兼容性【例如：Oracle不支持GUID的自增长列、支持字段名的长度最大为30】，所以要为不同的数据库创建不同的实体映射类且必须为单独的项目，这样才能通过配置文件加载指定的映射信息）

仓储

领域服务

工作单元

......

2、按模块+功能建立子目录，例如：BaseManage（基础管理）\User（用户管理）；

3、在子目录下创建相应的实体；

4、需要为每种数据库创建不同的实体映射类，且必须为单独项目——TG.UBP.Core.EntityConfiguration.XXX项目

5、在子目录下创建相应的领域服务类，在其中使用UOW开启事物、修改数据库、提交事物；

6、由于OSharp框架采用的是泛型仓储，所以一般不需要为每个实体创建单独的仓储；

7、工作单元是从属于泛型仓储的，也不需要单独创建；

**应用层:** 应用层提供一些应用服务（Application Services）方法供展现层调用。一个应用服务方法接收一个DTO(数据传输对象)作为输入参数，使用这个输入参数执行特定的领域层操作，并根据需要可返回另一个DTO。在展现层到领域层之间，不应该接收或返回实体(Entity)对象，应该进行DTO映射。一个应用服务方法通常被认为是一个工作单元（Unit of Work）。用户输入参数的验证工作也应该在应用层实现。ABP提供了一个基础架构让我们很容易地实现输入参数有效性验证。使用AutoMapper来进行实体与DTO之间的映射。

具体规则：

1、应用层主要包括三个内容：

服务接口

服务类

Dto

......

2、按模块+功能建立子目录，例如：BaseManage（基础管理）\User（用户管理）；

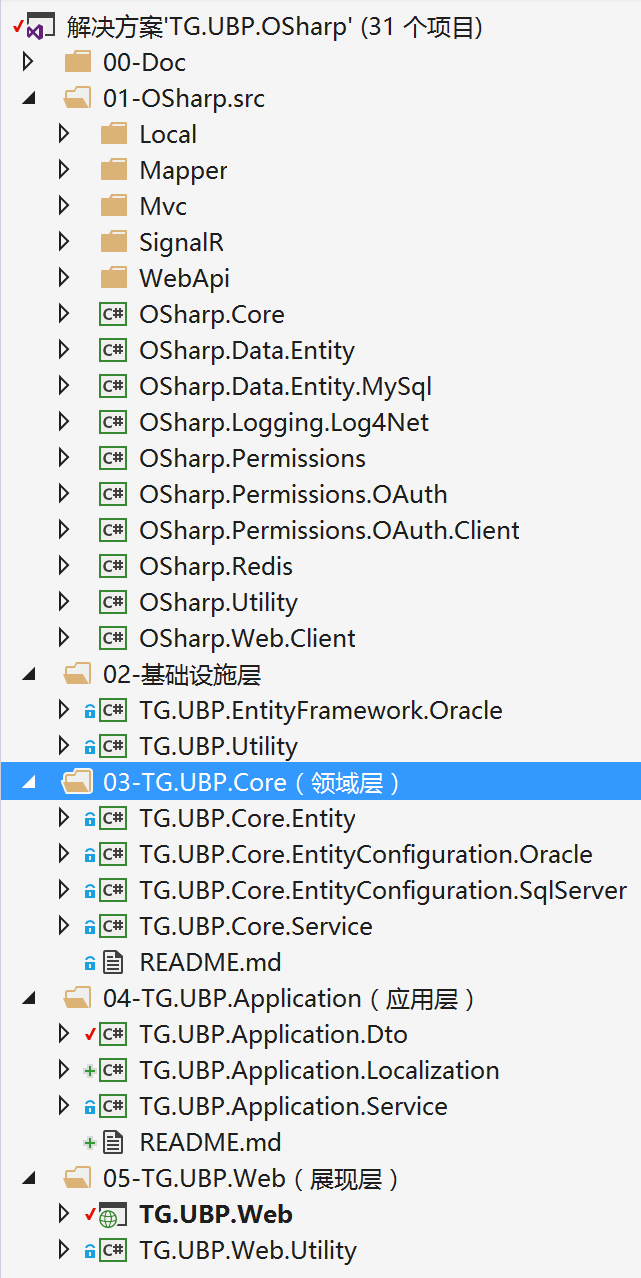
3、在子目录下创建相应的服务接口、服务类和Dto目录；

4、在Dto目录下创建需要的Dto类；

5、在服务类中要完成对Dto的数据验证；

**展现层**：Web层使用ASP.NET MVC和Web API来实现。JS框架使用了EasyUI+Jquery。

1. 项目结构：



OSharp框架源码

项目文档

系统工具库

仅仅Web层会用到的系统工具库

1. 源码服务器说明：

<http://www.cnblogs.com/zhaoyanjun/p/5882784.html>

源码服务器采用github，地址：<https://github.com/NC-ThreeGo/TG.UBP.OSharp>

每个人在github上注册自己的账号，然后在github的[NC-ThreeGo](https://github.com/NC-ThreeGo)组织中的Team中增加你们，并设置具有Write权限。

1. 问题：
2. 数据库的表名是否要按照下面的规则命名：模块名\_TXXX\_表名，例如：用户表-Base\_T001\_Users，模块表-Base\_T002\_Modules？（默认以实体类的类名作为表名）；
3. ID字段是否需要使用有意义的ID字段，例如：用户表ID-User\_ID，模块表ID-Module\_ID？（如果不使用有意义的ID字段，到时多表查询时ID字段会冲突）【实体类仍然使用Id属性，通过映射来实现】；