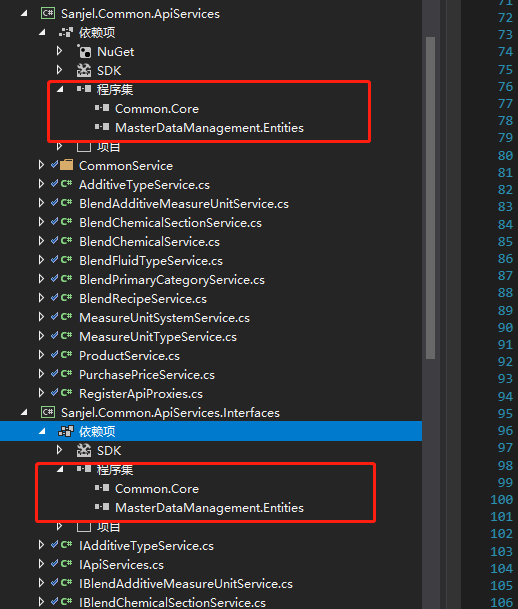
Eservices与MDM的集成

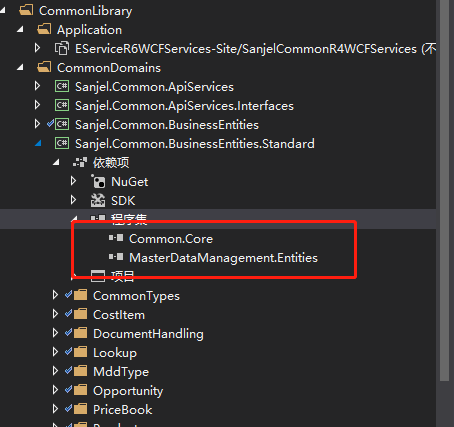
**一：Eservice与MDM之间的依赖关系**

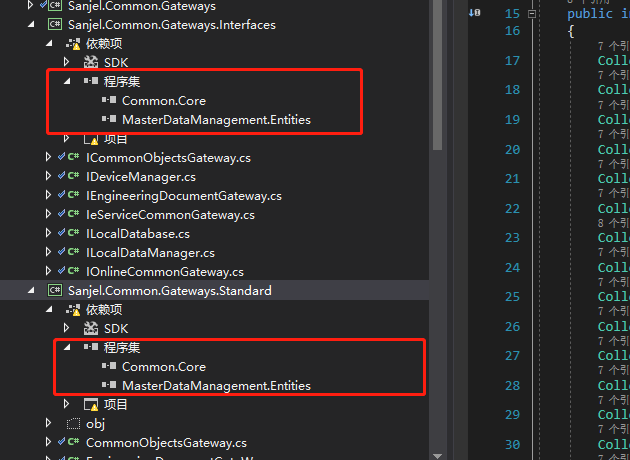
1. commonlibrary中对于MDM项目中实体的引用

1.1 首先肯定是扩展出来的api层。

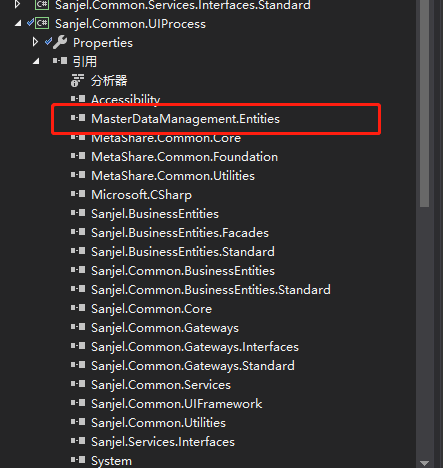
在sanjel.common.apiservice和sanjel.common.apiservices.interfaces中添加实体的dll.



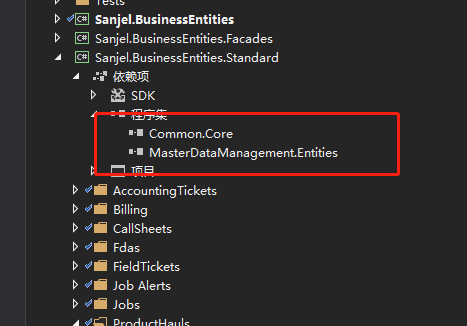
* 1. 然后是sanjel.common.businessentites.standard,因为这里面有一些关于计算cost，MDM实体与原有实体的转化等等
  2. Sanjel.Common.Gateways.Interfaces，Sanjel.Common.Gateways.Standard中也有引用，因为主要的onlineCommonObjectsGateway在这一层。



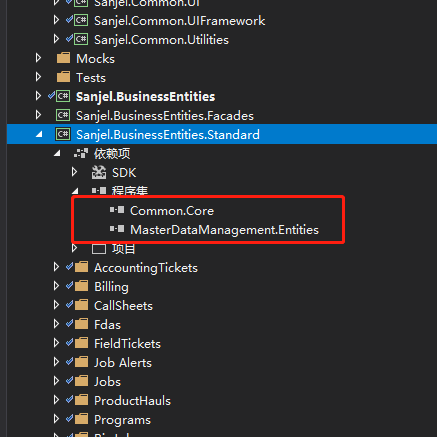
1.4 Sanjel.Common.UIProcess中也引用了，因为这层主要封装的都是一些winform中公共的基础设施。主要实在CommonApplicationController用到了MDM的实体。



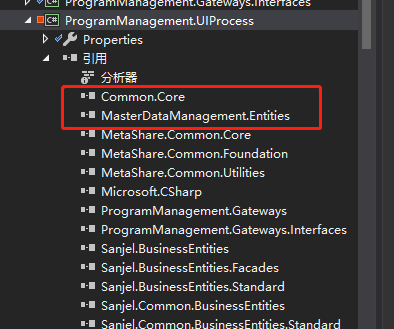
* 1. Sanjel.BusinessEntities.Standard引用了MDM中的实体dll



Commonlibrary中引用mdm实体的地方就这些。



1. 在SanjelProgramManagement项目中，需要在ProgramManagement.UIProcess中添加MDM实体引用。



后续如果需要用到API接口，先需要在项目中引用如下图2个dll进行开发。

如果需要调用



**二：原有eservice使用 MDM数据**

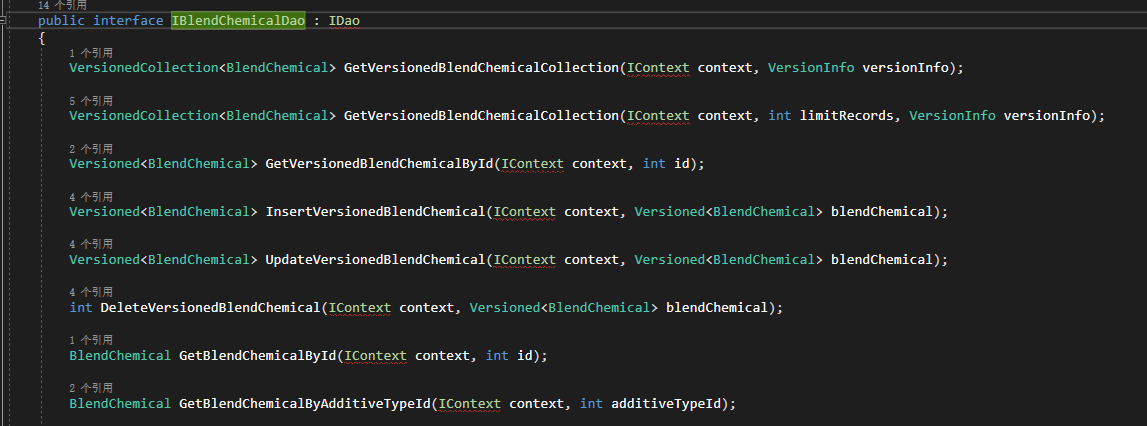
原有的MDM数据在eservice中虽然有版本的字段但没真正实现版本控制，而且没有数据的维护界面，比较难以管理。

MDM一般都是引用数据，在TestData中可以看到他所需要的所有初始化。数据库中也提供了表去存储。

例如：BlendChemical

以前的MDM数据就是通过IDAO定义实体所需要的接口，例如：

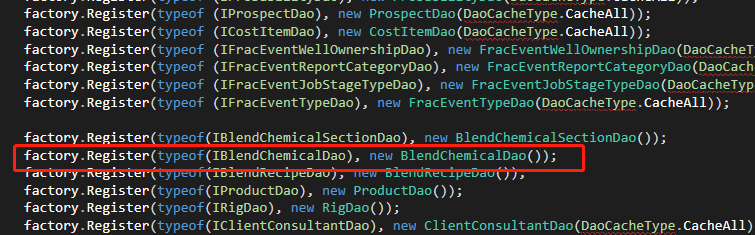
定义好IblendChemicalDao接口



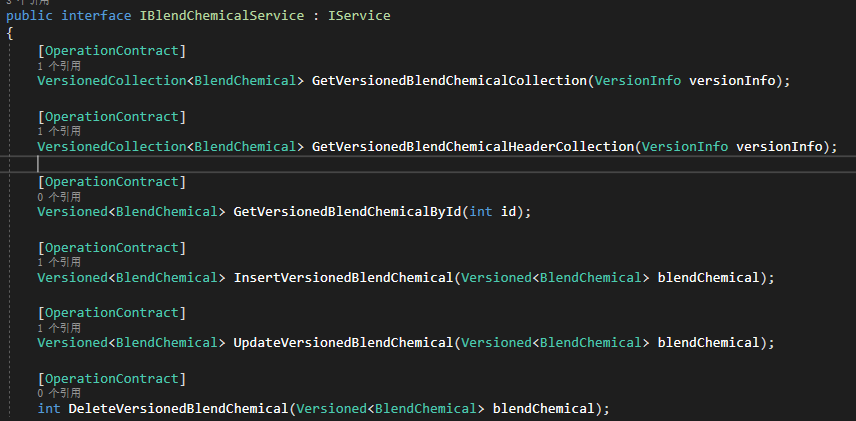
然后过blendChemicalDao实现IblendChemicalDao接口和继承EserviceDaobase（EserviceDaobase实现了基类DaoBase的基类方法）。



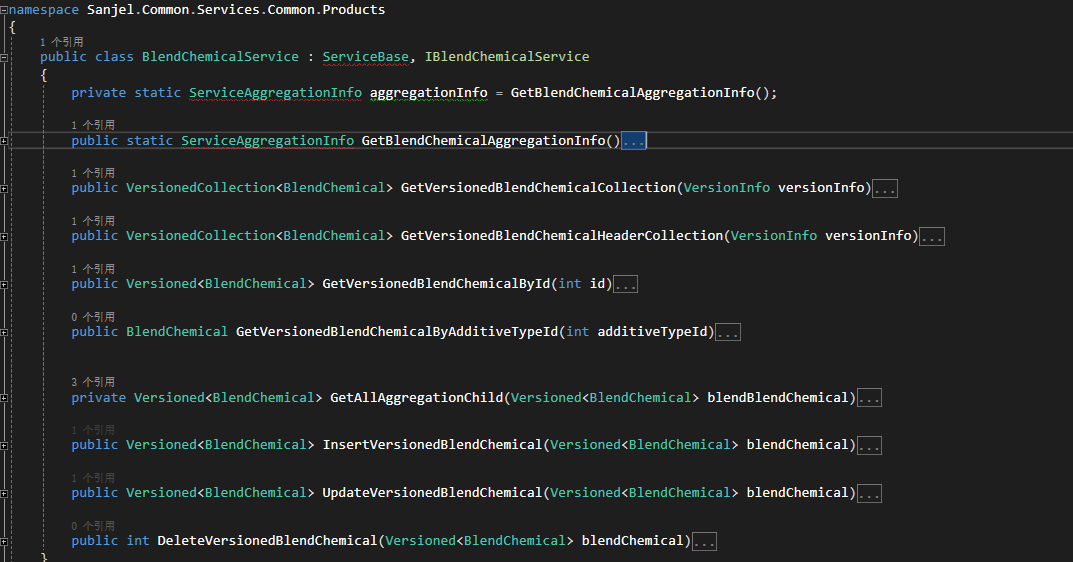
注册Dao到 DaoFactory(RegisterCommonDaos.cs)



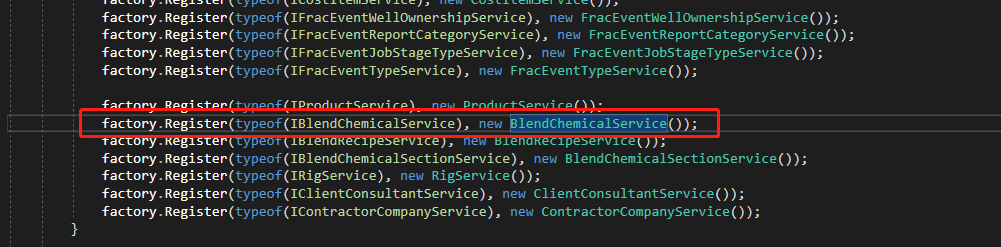
IblendChemicalService定义了业务层所需要的接口



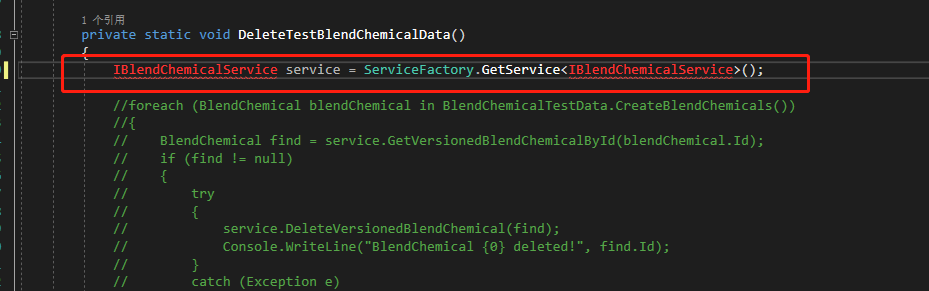
BlendChemicalService实现IblendChemicalService继承ServiceBase（ServiceBase实现了通用的service逻辑）



注册service到Factory(RegisterCommonServices.cs)



使用：从注册是厂里拿到你所需要的接口，很像.NetCore Mvc自带的依赖注入的使用方法。拿到具体service后就可以调用具体service中的具体方法去使用了。



**三：通过http通信以RESTful API原则使用MDM数据**

升级后的MDM数据都是通过以API接口的形式来完成数据的传输来替代offline模式加载MDM数据，这样做的目的不仅可以实现维护MDM数据以及控制MDM数据版本，也可以慢慢向微服务架构靠拢。

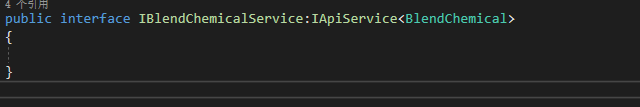
升级后的使用方式：

MDM项目会提供完整的实体的各种接口，Commonlibrary只需要定义好各个实体接收的接口即可。

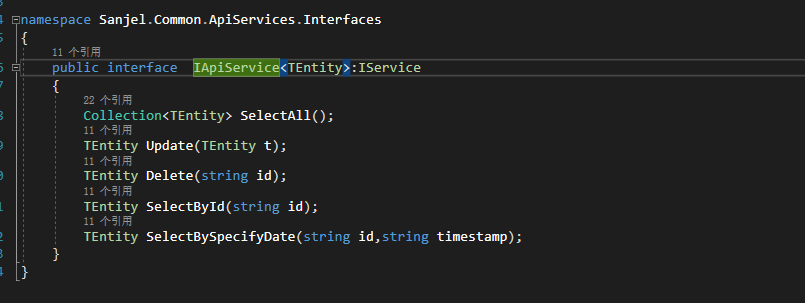
首先Commonlibrary中会有API.intrerface和API的实现类库。2个类库主要是定义每个实体所需要的接口和实现。需要引入MDM实体的dll

例如：BlendChemical

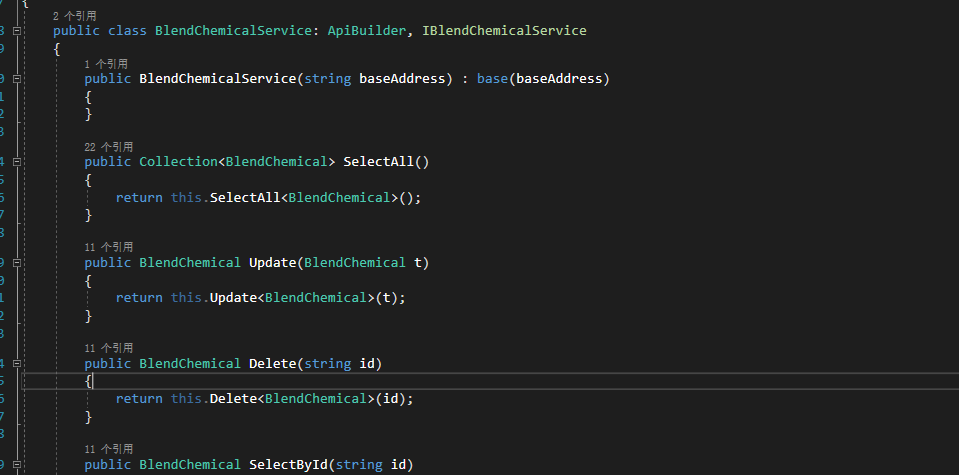
IblendChemicalService继承IapiService，如果有需要新的接口，在接口中定义。



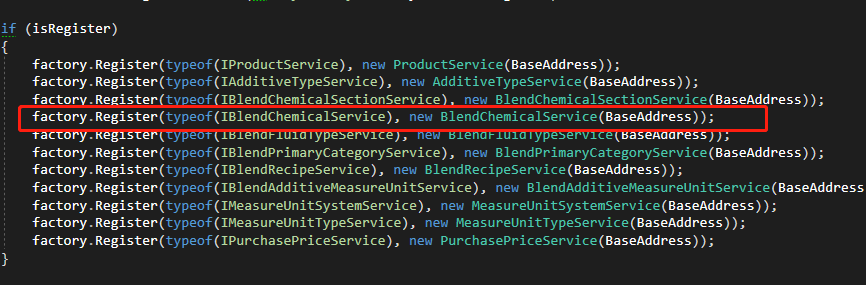
Iapiservice定义了常用的接口。



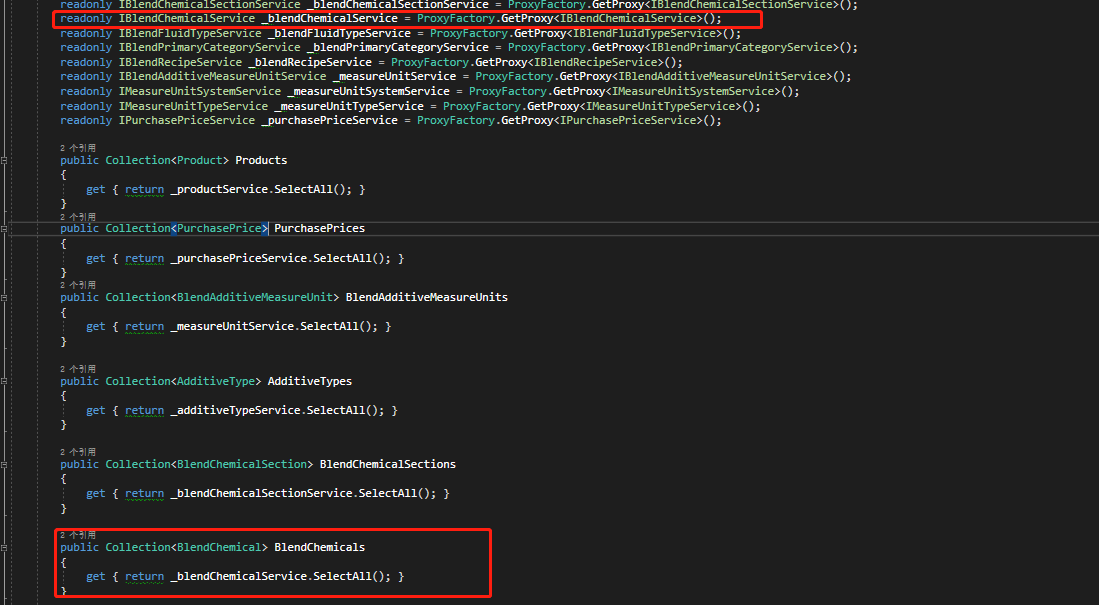
BlendChemicalService去实现接口中定义好的接口 ,apibuilder中实现了Iapiservice常用的接口，如果IblendChemicalService中定义了新的接口，只需要在BlendChemicalService单独实现即可。



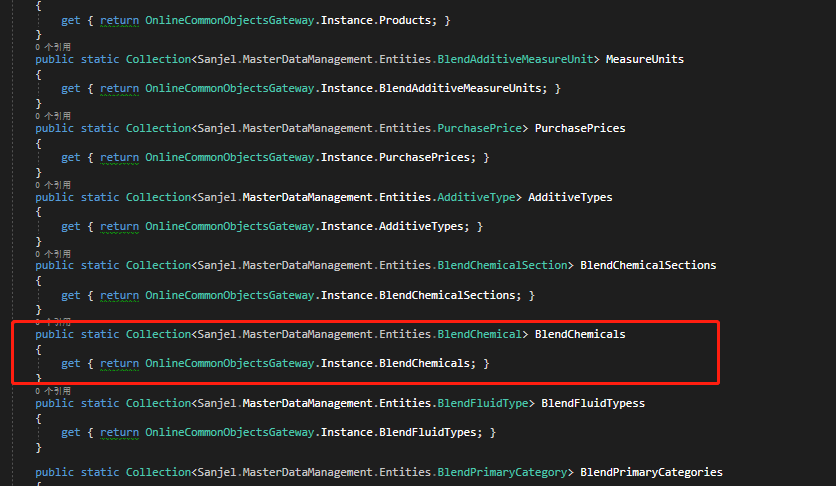
然后依旧是注册到factory里。不同的是这里多了baseAddress，baseaddress是MDM api项目访问地址。



目前所有MDM查询对象集合的方法可以在onlineCommonObjectsGateway中去完成。方便上层调用。

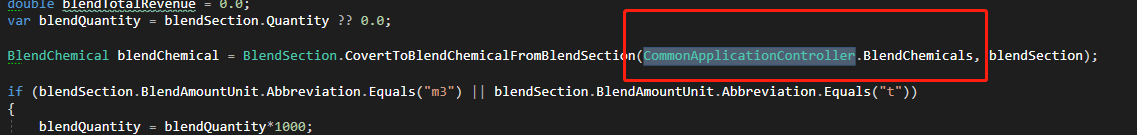


CommonApplicationController中拿到onlineCommonObjectsGateway中所有的MDM数据集合暴露到最外层 供其他项目调用。

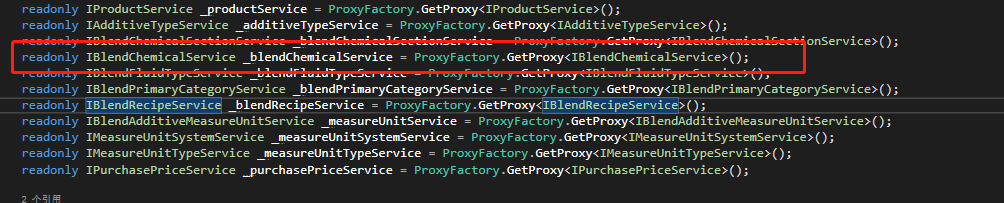


别的项目使用方法：

通过CommonApplicationController直接拿到具体实体的集合对象



或者通过如下图方法拿到service后调用所需方法即可。



MDM项目后续新加实体就要相对应的加入接口和实现去完善整个架构。整个实现如（二）所有步骤。