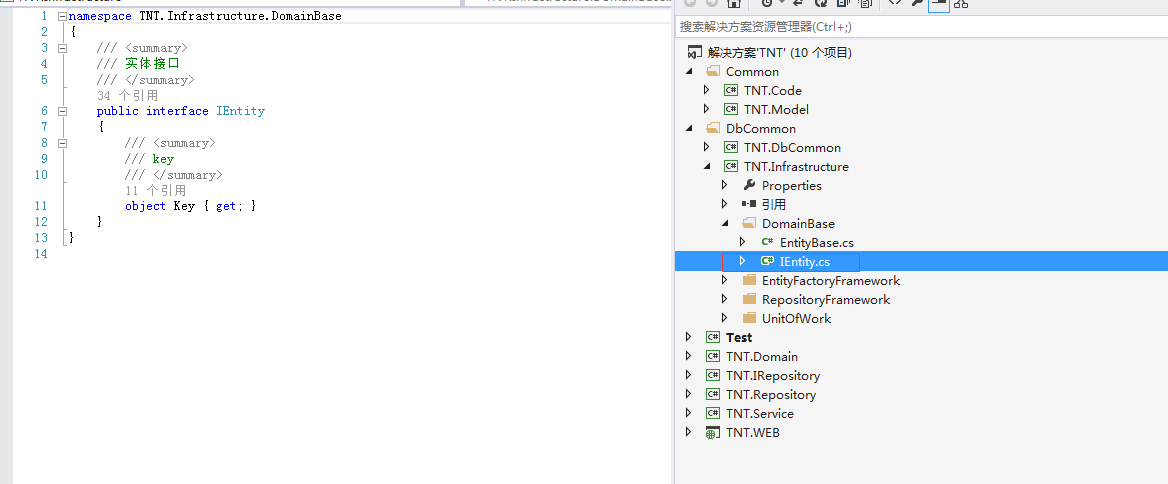
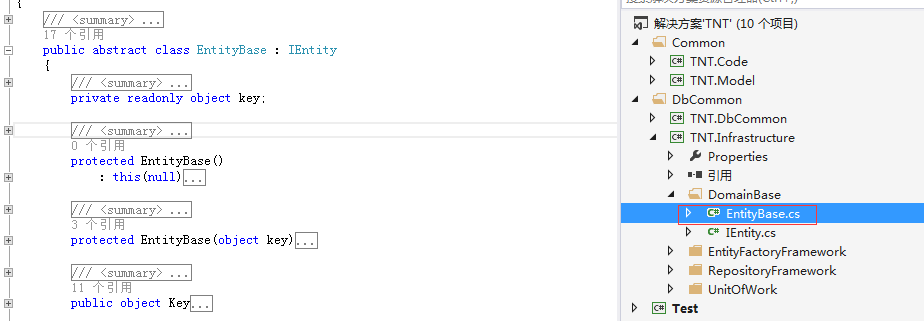
实体接口(这个是实体的总类型 相当于.NET Objcet 对象一样)



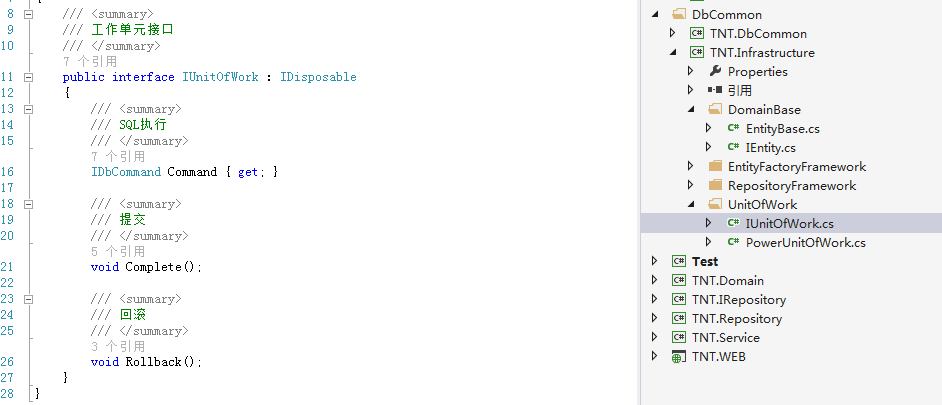
实体抽象 继承 实体接口



这类主要实现了一些基本方法重写等等

即所有的实体类 类型都应该是IEntityl类型 意思就是所以的实体类都应该继承 IEntity 接口

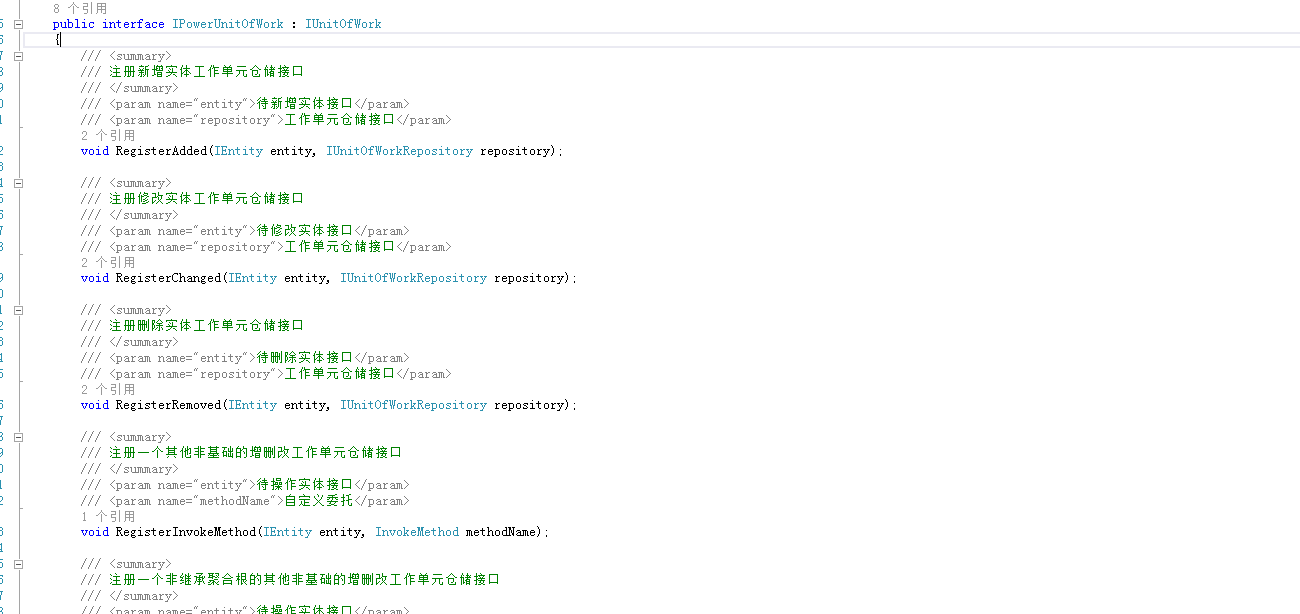
简单工作单元



简单工作单元原理: 相当于把数据库事物封装了到了一个对象里面 数据库链接 等不需要要自己管理 只需要 自己提交事物 或者 回滚事物

工作单元只提供 提交 和 回滚的方法 保证这个 工作单元 满足 一致性

委托工作单元



委托工作单元的原理: 把需要操作的方法签名注册到工作单元，工作单元使用字典表把需要操作的方法签名 和参数临时存储起来 提交工作单元的时候就去遍历这些字典 每个方法拿出来执行一次

优势：易扩展 ，易维护

劣势：不支持自增长ID， 和同一个实体对象操作多次(即对一张表 即增加 又修改) 如有类似的情况 可以使用简单工作单元

委托 工作单元 相当于 多了几个注册到工作单元 等待事物提交的方法

IEntity 要操作的实体 (因为所有的实体都是继承IEntity接口的)

IUnitOfWorkRepository 仓储接口(即数据库操作接口)



这个3个有点面向过程的 即匿名委托 和自定义委托   
这个3个不推荐使用   
比如根据ID 删除数据的 就会用到这个3个

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 工作单元比较 | 是否需要手动提交 | 是否需要手动回滚 | 是否需要自己处理异常 | 是否需要using | 自增长ID支持 |  |  |  |  |
| 简单工作单元 | 是 | 是 | 是 | 是 | 是 |  |  |  |  |
| 委托工作单元 | 是 | 否 | 否 | 是 | 否 |  |  |  |  |