一：RealProxyAOP方式

使用固定的写法，可以对所有的类中的方法调用时进行AOP前后处理

二：Unity框架实现AOP的方式

前期的准备工作

ILogger接口：

```
namespace Logging
{
    public interface ILogger
    {
        void Log(string value);
    }
}
```

ConsoleLogger类：

```
namespace Logging
{
    public class ConsoleLogger : ILogger
    {

        #region ILogger 成员

        public void Log(string value)
        {
            Console.WriteLine(String.Format("ConsoleLogger: {0}", value));
        }

        #endregion
    }
}
```

NullLogger类：

```
namespace Logging
{
    public class NullLogger : ILogger
    {
        #region ILogger 成员

        public void Log(string value)
        {
            // Do nothing
            Console.WriteLine("NullLogger: Hey, Nothing to do!");
        }

        #endregion
    }
}
```

实现步骤如下：

1. 建立容器；

在Unity中创建容器实例最简单的方法是直接使用构造函数创建，如下代码所示：

IUnityContainer container = new UnityContainer();

2. 将接口与类的映射注册到容器中；

在Unity中提供了一组Register方法供我们在容器中注册接口映射，如下代码所示：

container.RegisterType<ILogger, ConsoleLogger>();

3. 从容器中解析出正确的对象。

在Unity中提供了一组Resolve方法用以获取对象实例，如下代码所示

ILogger logger = container.Resolve<ILogger>();

三 。Autofac框架实现AOP.

四.使用Castle\DynamicProxy 实现动态代理来实现AOP

这里有一个关键点就是调用的 方法一定要是虚方法这个地方可以实现 AOP