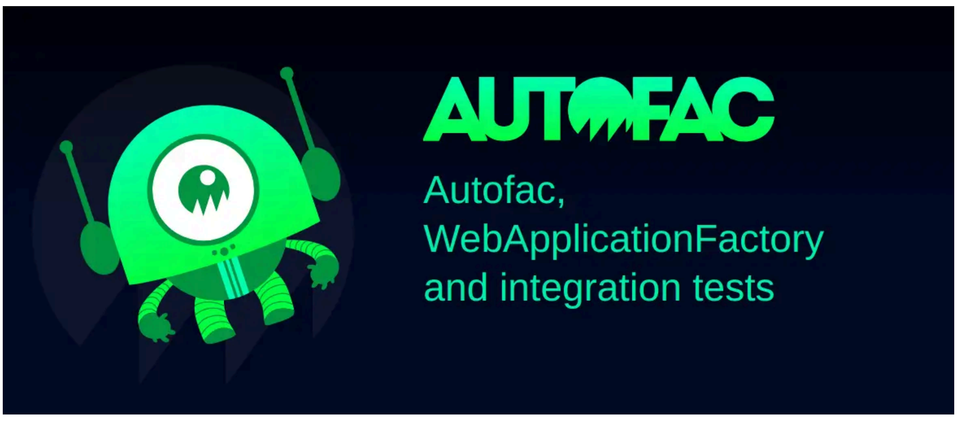
Autofac高级应用，一个接口多个实现类，你知道如何注册到容器并分别获取实例使用吗？

作者：微信公众号：【架构师老卢】 11-16 8:51 151



概述：当使用Autofac处理一个接口有多个实现的情况时，通常会使用键（key）进行区分或者通过IIndex索引注入，也可以通过IEnumerable集合获取所有实例，以下是一个具体的例子，演示如何在Autofac中注册多个实现，并通过构造函数注入获取指定实现。

当使用Autofac处理一个接口有多个实现的情况时，通常会使用键（key）进行区分或者通过IIndex索引注入，也可以通过IEnumerable集合获取所有实例，以下是一个具体的例子，演示如何在Autofac中注册多个实现，并通过构造函数注入获取指定实现。

首先，确保你已经安装了Autofac NuGet包：

```
Install-Package Autofac
```

然后，我们看一个示例：

```
using System;
using Autofac;

// 定义接口
public interface IService
{
    void Execute();
}

// 实现接口的两个类
public class ServiceA : IService
{
    public void Execute()
    {
        Console.WriteLine("ServiceA is executing.");
    }
}

public class ServiceB : IService
{
    public void Execute()
    {
        Console.WriteLine("ServiceB is executing.");
    }
}

// 包含多个实现的类
public class ServiceConsumer
{
    private readonly IService _service;

    //KeyFilterc通过Keyed方法获取指定IService实现实例
    //通过IIndex索引注入
    //IEnumerable<IService>获取所有IService的实现实例
    public ServiceConsumer(
      [KeyFilter("ServiceA")] IService serviceA,//通过KeyFilter键注入
      [KeyFilter("ServiceB")] IService serviceB,//通过KeyFilter键注入
      IIndex<ServiceA, IService> serviceA1,//通过IIndex索引注入
      IIndex<ServiceB, IService> serviceB1,//通过IIndex索引注入
      IEnumerable<IService> services//注入全部实现
    )
    {
        //根据实现选择进行选择赋值
        _service=serviceA;
        _service=serviceB;
        _service=serviceA1;
        _service=serviceB1;
    }

    public void UseService()
    {
        _service.Execute();
    }
}

class Program
{
    static void Main()
    {
        // 创建 Autofac 容器构建器
        var builder = new ContainerBuilder();

        // 注册多个实现，使用 Keyed 注册方式
        builder.RegisterType<ServiceA>().Keyed<IService>("ServiceA");
        builder.RegisterType<ServiceB>().Keyed<IService>("ServiceB");

        // 注册 ServiceConsumer 类
        builder.RegisterType<ServiceConsumer>();

        // 构建容器
        var container = builder.Build();

        // 通过构造函数注入获取指定实现
        var serviceConsumer = container.Resolve<ServiceConsumer>();
        serviceConsumer.UseService();
    }
}
```

这个示例中：

1. 定义了 `IService` 接口和两个实现类 `ServiceA` 和 `ServiceB`。
2. 使用 `Keyed` 注册方式，为每个实现指定了一个键。
3. 创建了 `ServiceConsumer` 类，通过构造函数注入了 `IService` 实例，使用了 `KeyFilter` 特性指定了要注入的实现。
4. 注册 `ServiceConsumer` 类，Autofac 将自动解析构造函数，并注入指定的实现。

通过这样的方式，你可以在构造函数中注入指定键对应的实现，实现了一个接口有多个实现时的注册和获取。