C#实例：[1]Unity依赖注入使用

听语音

* |
* 浏览：1726
* |
* 更新：2015-01-13 10:34
* |
* 标签：[it](http://jingyan.baidu.com/tag?tagName=it)
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=1)1
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=2)2
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=3)3
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=4)4
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=5)5
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=6)6
* [](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=7)7

[分步阅读](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html)

[一键约师傅](http://shifu.baidu.com/service/YPSJ?fr=exp-qb-pc&cityCode=131" \t "_blank)

**[百度师傅升级硬盘，飞一般的游戏体验](http://shifu.baidu.com/service/YPSJ?fr=exp-qb-pc&cityCode=131" \t "_blank)**

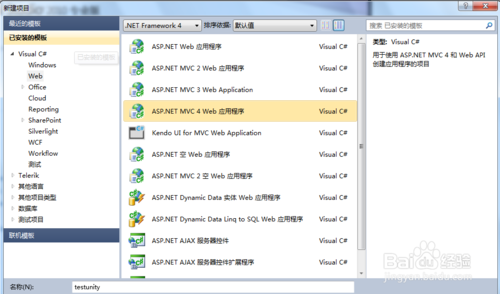
Unity是一个轻量级的可扩展的依赖注入容器，支持构造函数，属性和方法调用注入。Unity可以处理那些从事基于组件的软件工程的开发人员所面对的问题。构建一个成功应用程序的关键是实现非常松散的耦合设计。下面介绍一下c#中使用unity的方法（我是以webapi项目为例，但本例中并没有针对webapi做特殊处理）

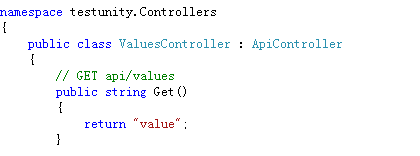
工具/原料

* vs2010

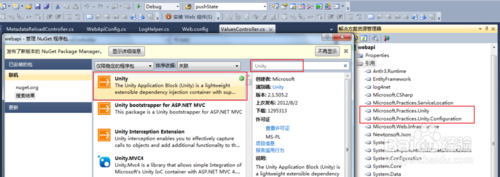
方法/步骤

1. 新建一个mvc4 webapi项目，下面的例子只用到get方法

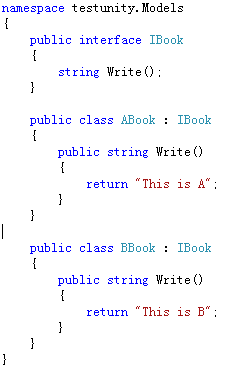
[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=1)

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=2)

1. 用nuget安装unity，如图

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=3)

1. 新建一个接口类，以及继承该接口的两个类

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=4)

1. 直接在api/values的get中尝试简单实现unity

using (IUnityContainer container = new UnityContainer())

            {

                container.RegisterType<IBook, BBook>();

                IBook a = container.Resolve<IBook>();

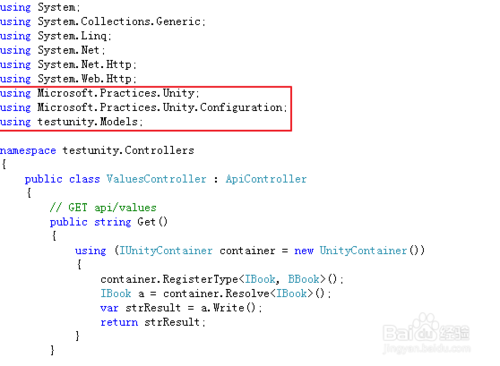
                var strResult = a.Write();

                return strResult;

            }

然后在浏览器中查看，页面显示的返回值，是BBook的

如果container.RegisterType<IBook, BBook>();中的BBook改为ABook，返回值就是ABook的内容

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=5)

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=6)

1. config中代码如下

在configSections中加入

<section name="unity"

type="Microsoft.Practices.Unity.Configuration.UnityConfigurationSection, Microsoft.Practices.Unity.Configuration" />

在configuration中加入

<unity  xmlns="http://schemas.microsoft.com/practices/2010/unity">

    <container>

      <register type="testunity.Models.IBook,testunity" mapTo="testunity.Models.ABook, testunity" />

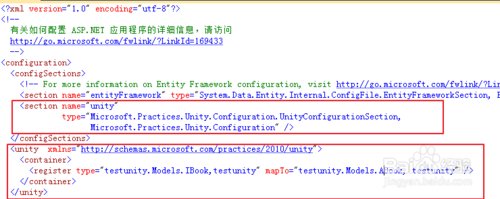
    </container>

  </unity>

注意 type="testunity.Models.IBook,testunity" mapTo="testunity.Models.ABook, testunity"

testunity.Models.IBook是命名空间加类名

testunity是程序集的名称

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=7)

1. 然后cs的代码改成

using (IUnityContainer container = new UnityContainer())

{

    UnityConfigurationSection configuration = (UnityConfigurationSection)ConfigurationManager.GetSection("unity");

    configuration.Configure(container);

    IBook a = container.Resolve<IBook>();

    var strResult = a.Write();

    return strResult;

}

在浏览器中可以看到返回结果对应的是config中register的那个类

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=8)

1. 每次调用都要写IUnityContainer container = new UnityContainer()显然不是好办法

那就把container封装到一个单列类中

简单实现如下

public class ServiceLocator:IServiceProvider

    {

        private readonly IUnityContainer \_container;

        private static readonly ServiceLocator instance = new ServiceLocator();

        private ServiceLocator()

        {

            UnityConfigurationSection section = (UnityConfigurationSection)ConfigurationManager.GetSection("unity");

            \_container = new UnityContainer();

            section.Configure(\_container);

        }

        public static ServiceLocator Instance

        {

            get { return instance; }

        }

        public object GetService(Type serviceType)

        {

            return \_container.Resolve(serviceType);

        }

        public T GetService<T>()

        {

            return \_container.Resolve<T>();

        }

    }

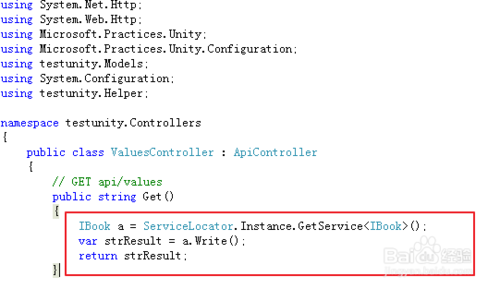
cs代码修改如下

IBook a = ServiceLocator.Instance.GetService<IBook>();

var strResult = a.Write();

return strResult;

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=9)

[](http://jingyan.baidu.com/album/c74d6000840b260f6b595d78.html?picindex=10)