写了一个AsyncRestService

可以在Silverlight端直接使用API方式调用服务端Restful Service，而不是使用原始的RestClient，使用方式类似于传统的WCF API调用，如下：

先建一个Contract的dll工程，定义Contract：

    [ServiceContract]

    public interface ICustomerService

    {

        [WebInvoke(UriTemplate = "/Update", RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json, Method = "PUT", BodyStyle = WebMessageBodyStyle.Wrapped)]

        void Update(int id, Customer customer);

        [WebGet(UriTemplate = "/GetCustomer/{id}", RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json)]

        Customer GetCustomer(string id);

    }

    [DataContract]

    public class Customer

    {

        。。。。

}

真实的Service：

    public class CustomerService : ICustomerService

    {

        。。。。。

}

通过WebHost发布，在Global.asax里注册服务：

RouteTable.Routes.Add(new ServiceRoute(typeof(ICustomerService).Name, new WebServiceHostFactory(),typeof(CustomerService).));

------------------------------- 服务端到此结束----

下面是客户端使用方式：

1.添加一个客户端的Contract工程，添加服务端Contract文件的Link引用，该工程需要引用AsyncRestService的dll（因为Silverlight那边没有定义WebGet和WebInvoke等，为了编译通过，我在AsyncRestService的dll里定义了相关类型，类名和命名空间都和.net framework里的一样）

2. 调用方法：

            AsyncRestService<ICustomerService> service = new AsyncRestService<ICustomerService>("<http://localhost:3333/>");  // 有重载方法，可以传入Loading的handler和ErrorMessage的handler

            service.NoCallback.Update(1, new Customer() { …… });  // 没有返回值的

            service.Callback<Customer>(x =>

            {

                Console.WriteLine(x.ReturnData.Name);

            }).GetCustomer("11111");   // 有返回值的，需要定义回调

// Service的同步调用的方式还没做，但也是应该也是可以实现的