**[WebApi系列~自主宿主HttpSelfHost的实现](http://www.cnblogs.com/lori/p/4120428.html)**

[回到目录](http://www.cnblogs.com/lori/p/4500882.html)

宿主一词我们不会陌生，它可以看作是一个基础设施，它为一些服务和功能提供最底层的支持，如你的web应用程序可以运行在iis或者apache上，而这两个东西就是web应用程序的宿主，而今天说的自主宿主SelfHost就是说，它可以自己去监听自己的服务，如你可以把一个web应用程序宿主到一个console控制台程序上，或者把一个webApi宿主到一个console或者windowService上，这都是可以的。

**一　需要添加一些程序集引用**

**二　代码实现**

[复制代码](javascript:void(0);)

#region Web Api监听  
 Assembly.Load("Lind.DDD.TestApi");  //手工加载某个api程序集的controller

var config = new HttpSelfHostConfiguration("http://localhost:3333");

config.Routes.MapHttpRoute("default", "api/{controller}/{id}", new { id = RouteParameter.Optional });

var server = new HttpSelfHostServer(config);

server.OpenAsync().Wait();

Console.WriteLine("Server is opened");

#endregion

[复制代码](javascript:void(0);)

**三   web api代码**

[复制代码](javascript:void(0);)

/// <summary>

/// 测试webapi

/// </summary>

public class TestController : ApiController

{

// GET api/<controller>

public IEnumerable<string> Get()

{

return new string[] { "value1", "value2" };

}

// GET api/<controller>/5

public string Get(int id)

{

return "value";

}

// POST api/<controller>

public void Post([FromBody]Demo value)

{

Thread.Sleep(10000);

Logger.Core.LoggerFactory.Instance.Logger\_Info(value.ToString());

}

// PUT api/<controller>/5

public void Put(int id, [FromBody]string value)

{

}

// DELETE api/<controller>/5

public void Delete(int id)

{

}

}

public class Demo

{

public string appName { get; set; }

public string url { get; set; }

public override string ToString()

{

return string.Format("appName:{0},url:{1},datetime:{2}", this.appName, this.url, DateTime.Now);

}

}

[复制代码](javascript:void(0);)

**四   测试**