[WebAPI用法](http://www.cnblogs.com/youring2/archive/2013/03/08/2949455.html)

昨天看了两篇文章：

[ASP.NET Web API（一）：使用初探，GET和POST数据](http://www.cnblogs.com/parry/archive/2012/09/27/ASPNET_MVC_Web_API.html)【[Parry](http://home.cnblogs.com/u/parry/)】

[HttpClient + ASP.NET Web API, WCF之外的另一个选择](http://www.cnblogs.com/dudu/archive/2012/05/11/asp_net_webapi_httpclient.html)【[dudu](http://home.cnblogs.com/u/dudu/)】

通过这两篇文章让我了解了WEB API的简单用法。

Parry是在MVC中使用WebAPI：在Visual Studio 2012中新建MVC4项目，在App\_Start目录下有一个WebApiConfig.cs文件，这个文件中就是相应的Web API的路由配置了。

我也根据这两篇文章写了一个简单的测试程序。

先创建了一个UserModel

public class UserModel

{

public string UserID { get; set; }

public string UserName { get; set; }

}

然后添加Web API Controller

public class UserController : ApiController

{

public UserModel getAdmin()

{

return new UserModel() { UserID = "000", UserName = "Admin" };

}

}

注册路由

public static void Register(HttpConfiguration config)

{

config.Routes.MapHttpRoute(

name: "DefaultApi",

routeTemplate: "api/{controller}/{id}",

defaults: new { id = RouteParameter.Optional }

);

}

在Global中注册

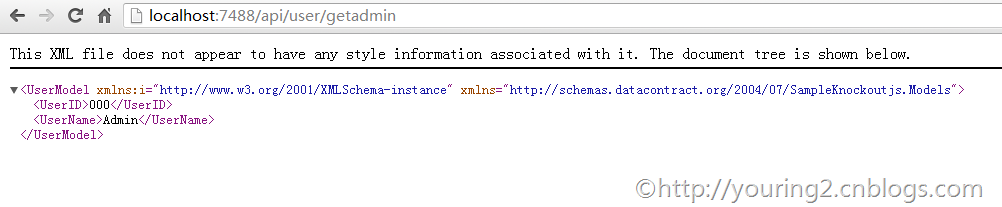
protected void Application\_Start(object sender, EventArgs e)

{

WebApiConfig.Register(GlobalConfiguration.Configuration);

}

这个时候用地址栏访问地址：api/user/getadmin



这个时侯默认返回的是XML数据模型。

使用AJAX请求这个api，指定数据格式为json

$.ajax({

type: 'GET',

url: 'api/user/getadmin',

dataType: 'json',

success: function (data, textStatus) {

alert(data.UserID + " | " + data.UserName);

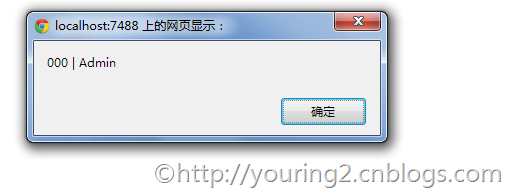
},

error: function (xmlHttpRequest, textStatus, errorThrown) {

}

});

alert出来的结果是：



这样看来，真的是dudu所说的，可以根据请求的数据类型返回指定的数据格式。

**POST数据**

修改一下controller，添加一个add方法

public bool add(UserModel user)

{

return user != null;

}

只为了测试，所以这里只判断一下传入的实体是否为空，如果不为空则返回true

我在页面上添加了一个button，代码如下：

<input type="button" name="btnOK" id="btnOK" value="发送POST请求" />

添加JS代码

$('#btnOK').bind('click', function () {

//创建ajax请求，将数据发送到后台处理

var postData = {

UserID: '001',

UserName: 'QeeFee'

};

$.ajax({

type: 'POST',

url: 'api/user/add',

data: postData,

dataType: 'json',

success: function (data, textStatus) {

alert(data);

},

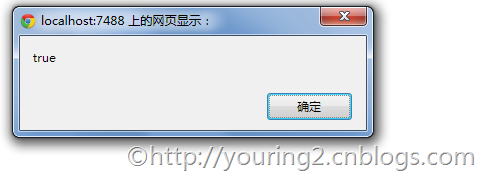
error: function (xmlHttpRequest, textStatus, errorThrown) {

}

});

});

再次运行页面



我们附加进程进行调试，在发送ajax请求的时候，服务器段接收到的数据如图：

