WebApi

1. 新建空web项目
2. 添加webApi 2控制器
3. 访问时候需要添加 /Api/Controller Get开头的方法名就会自动匹配httpmethod=get,参数的来源可以使用特性来标记 FromBody /FromUri 如果Get方法的参数是模型对象则需在该参数前标注 FromUri
4. webapi返回类型会根绝客户端需求自动处理返回成xml或者json等类型
5. 借助Fiddler/PostMan工具调试webapi接口 推荐postman下载地址: <https://www.getpostman.com>
6. httpmethod 类型:get(查)/post(增)/put(改)/delete(删)
7. Post请求时候如果请求体是 name=zhangsn&age=8这样的格式,如果使用 string AddNew(string phoneNum, string username)这种普通参数会有很多坑(参考C#进阶系列—WebApi接口参数不在困惑:传参详解),所以不要用.Post请求时候都是用模型对象,并标注FromBody
8. Post请求时候如果contenttype=application/json那么报文体就的是json格式
9. webapi中可以通过[Route]和[RoutePrefix]来自定义路由,RoutePrefix作用于controller,route作用于action,一旦设置了,默认路由将会不起作用[Route(“GetById2”)]访问时候路径就是 /GetById2 如果是给controller配置则需要在 /{RoutePrefix}/{Route},利用Route获取参数 [Route(“Login/{username}/{password}”)]可以获取username和password为参数
10. 多版本共存

1>配置WebApiConfig路由

config.Routes.MapHttpRoute(

name: "DefaultApi1",

routeTemplate: "api/v1/{controller}/{id}",

defaults: new { id = RouteParameter.Optional }

);

config.Services.Replace(typeof(IHttpControllerSelector),new VersionControllerSelector(config));

2>添加VersionControllerSelector类

VersionControllerSelector类代码:

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Text.RegularExpressions;

using System.Web.Http;

using System.Web.Http.Controllers;

using System.Web.Http.Dispatcher;

namespace WebApi

{

public class VersionControllerSelector:DefaultHttpControllerSelector

{

private HttpConfiguration \_config ;

public VersionControllerSelector(HttpConfiguration configuration) : base(configuration)

{

this.\_config = configuration;

}

public override IDictionary<string, HttpControllerDescriptor> GetControllerMapping()

{

Dictionary<string , HttpControllerDescriptor> dictionary =new Dictionary<string, HttpControllerDescriptor>();

foreach (var asm in \_config.Services.GetAssembliesResolver().GetAssemblies())//遍历获取的所有的程序集

{

//找到继承自该程序集并且非抽象类集合

var controllerTypes= asm.GetTypes().Where(t => t.IsAbstract == false && typeof(ApiController).IsAssignableFrom(t));

foreach (var type in controllerTypes)

{

string ctrlTypeNameSpace = type.Namespace;

if (ctrlTypeNameSpace != null)

{

var match=Regex.Match(ctrlTypeNameSpace, @"\.V(\d)");

if (!match.Success)

{

continue;

}

string verNum = match.Groups[1].Value; //把版本号提取出来

string contTypeName = type.Name; //把PersonController取出来

var matchController = Regex.Match(contTypeName, "(.+)?Controller$");

if (!matchController.Success)

{

continue;

}

string ctrlName = matchController.Groups[1].Value.ToLower();//得到 person

string key = ctrlName + "v" + verNum;

dictionary[key]=new HttpControllerDescriptor(\_config,ctrlName,type);

}

}

}

return dictionary;

}

public override HttpControllerDescriptor SelectController(HttpRequestMessage request)

{

string controller= request.GetRouteData().Values["controller"].ToString().ToLower();//获取请求的控制器

var ctrlMpping = GetControllerMapping();//获取匹配的控制器数组

var matchVer = Regex.Match(request.RequestUri.PathAndQuery, @"/v(\d+)/");//检测请求地址是否匹配

if (!matchVer.Success)//匹配失败

{

return base.SelectController(request);

}

string verNum = matchVer.Groups[1].Value;//获取版本号2

string key = controller + "v" + verNum;//获取键值对中的key

if (ctrlMpping.ContainsKey(key))//判断字典是否包含该key

{

return ctrlMpping[key];//返回应该匹配的控制器

}

return base.SelectController(request);

}

}

}

1. WebApi的Filter:如果项目也添加了mvc的程序集引用,那么一定要用System.Web.Http.Filters的IAuthorizationFilter
2. IAuthorizationFilter,报文头的UserName为Admin则验证通过
3. 我为Action打上了AllowAnonymousAttribute，验证逻辑就放过了这个API而不进行权限校验。

JWT

1. net版本4.6.2以上
2. 文档地址:
3. <http://www.cnblogs.com/ldybyz/p/6943827.html>
4. <http://www.jianshu.com/p/576dbf44b2ea>
5. <http://www.jianshu.com/p/180a870a368a>
6. Jwt由三块组成 header /payload/signature,可以吧用户名等保存在payload中,payload和header是base64编码,可以轻松解码.payload有个exp的值,为过期时间和1970/1/1 00:00:00的相差秒数,计算时候应该使用utc(世界标准)时间DateTime.UtcNow.AddSeconds(10);
7. 检测接口请求时间和服务器时间是否超过一定时间,超过则请求无效