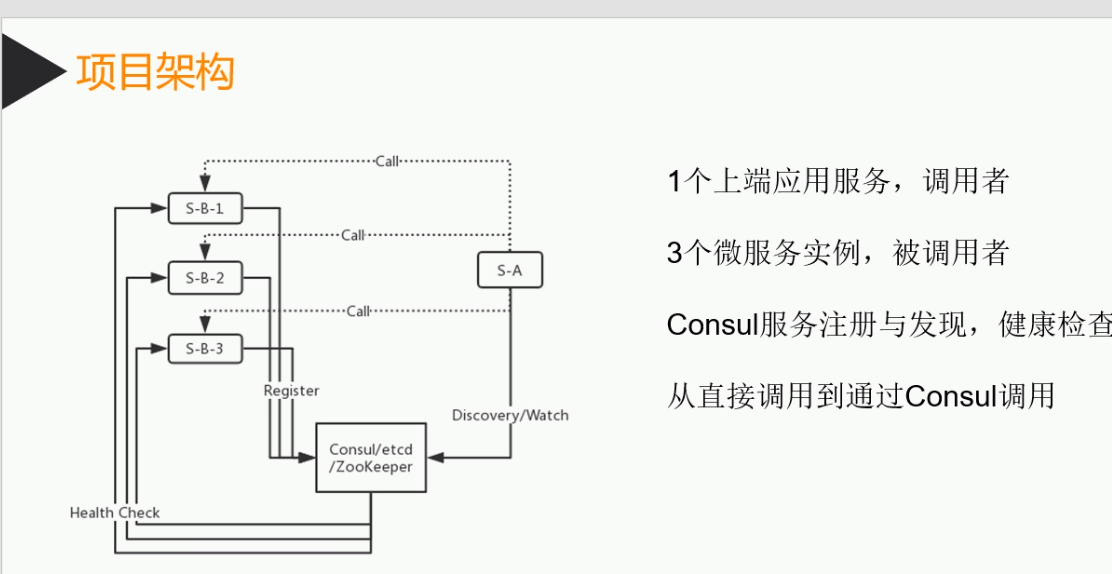
Consul(etcd/ZooKeeper): 服务注册与发现，心跳检查. 将需要的服务注册进 Consul,可以动态 增加或者减少服务，Consul的心跳检测机制，可以检测该服务是否可用，不可用就自动下线该服务,主要用于集群

只是服务的管理，没有实现负载均衡，没有解决客户端和服务调用的的复杂性问题。

调用流程是 A 先到consul中，查看哪些 B可以用，consul告诉A以后， A再调用可以用的B



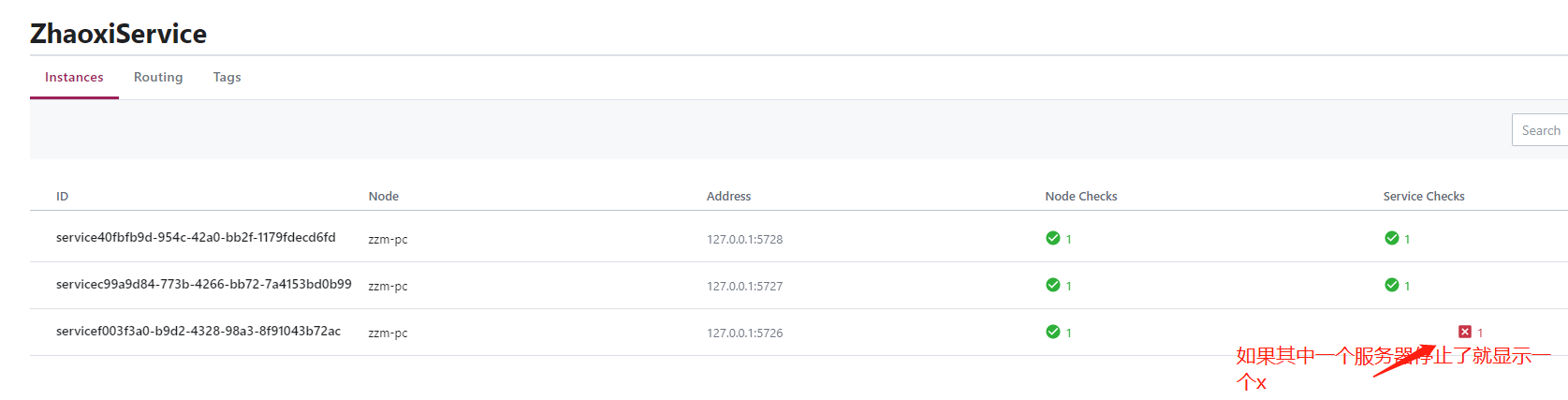
启动方法：

1. 启动多个api服务
2. 在consul目录中

Cmd: consul.exe agent –dev

显示Consul agent running! 就是成功了

浏览器输入：<http://localhost:8500/>



代码：

1. 在 serviceAPI项目中，添加 consul包
2. 添加consul帮助类

public static class ConsulHelper

{

public static void ConsulRegist(this IConfiguration configuration)

{

ConsulClient client = new ConsulClient(c =>

{

c.Address = new Uri("http://localhost:8500/");//这个是consul的地址

c.Datacenter = "dc1";//consul名称，可以自定义

});

string ip = configuration["ip"];//这个是当前网址，自动传进来的的，需要再startup中配置 this.Configuration.ConsulRegist(); 这句话

int port = int.Parse(configuration["port"]);//命令行参数必须传入

int weight = string.IsNullOrWhiteSpace(configuration["weight"]) ? 1 : int.Parse(configuration["weight"]); //权重 命令行中自定义的参数

//服务注册

client.Agent.ServiceRegister(new AgentServiceRegistration()

{

ID = "service" + Guid.NewGuid(),//唯一的--- 日志服务1,日志服务2等等

Name = "ZhaoxiService",//组名称-Group 支付服务，日志服务等等

Address = ip,//其实应该写ip地址

Port = port,//不同实例

Tags = new string[] { weight.ToString() },//标签

Check = new AgentServiceCheck()

{

Interval = TimeSpan.FromSeconds(12),//间隔12s一次

HTTP = $"http://{ip}:{port}/Api/Health/Index",//调用这个api接口

Timeout = TimeSpan.FromSeconds(5),//检测等待时间

DeregisterCriticalServiceAfter = TimeSpan.FromSeconds(60)//失败后多久移除该serviceAPI项目,最小值60S

}

});

//命令行参数获取

// Console.WriteLine($"{ip}:{port}--weight:{weight}");

}

}

1. Startup类的Configure中添加一句话

//只执行一次的服务注册，发现

this.Configuration.ConsulRegist();

1. 将serviceApi添加到 consul中

public IActionResult Index()

{

// base.ViewBag.Users = \_IUserService.UserAll();

// string url = "http://localhost:5726/api/users/all";

//api项目的接口地址

// 这是个集群，有多个api接口，执行相同的操作，例如 <http://localhost:5726/api/users/all>，<http://localhost:5727/api/users/all>等等，其中一个挂掉了，在心跳检测只有，consul就自动下线这个api 。

string url = "http://ZhaoxiService/api/users/all";

#region consul

ConsulClient client = new ConsulClient(c =>

{

c.Address = new Uri("http://localhost:8500/");//这个是consul的地址

c.Datacenter = "dc1";//consul名称，可以自定义

});

var response = client.Agent.Services().Result.Response;

//获取所有的ip+port+ 服务注册时的组名称ZhaoxiService(支付服务，日志服务等等)

//foreach (var item in response)

//{

// Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

// Console.WriteLine(item.Key);

// var service = item.Value;

// Console.WriteLine($"{service.Address}--{service.Port}--{service.Service}");

// Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

//}

Uri uri = new Uri(url);

var groupname = uri.Host;

//获取服务注册时的组名称ZhaoxiService

var serviceDictionary = response.Where(s => s.Value.Service.Equals(groupname, StringComparison.OrdinalIgnoreCase)).ToArray();

//url = $"{uri.Scheme}://{serviceDictionary.FirstOrDefault().Value.Address}:{serviceDictionary.FirstOrDefault().Value.Port}/api/users/all";

//负载均衡策略

AgentService agentService = null;

//平均策略，在大项目中有很多实例，随机策略就是平均策略

//DateTime.Now.Millisecond+ iSeed++ 保证绝对随机

//agentService = serviceDictionary[new Random(DateTime.Now.Millisecond + iSeed++).Next(0, serviceDictionary.Length)].Value;

//轮训策略（完全平均,不常用）

// agentService = serviceDictionary[iSeed++ % serviceDictionary.Length].Value;

//权重（常用）,权重是1，就在数组中添加一次， 权重是5，就在数组中添加5次，最后在数组里随机获取就行了

List<KeyValuePair<string, AgentService>> pairlist = new List<KeyValuePair<string, AgentService>>();

foreach (var item in serviceDictionary)

{

int count = int.Parse(item.Value.Tags[0]);

for (int i = 0; i < count; i++)

{

pairlist.Add(item);

}

}

agentService = pairlist.ToArray()[new Random(iSeed++).Next(0, pairlist.Count())].Value;

url = $"{uri.Scheme}://{agentService.Address}:{agentService.Port}/api/users/all";

#endregion

string content = InvokeApi(url);

base.ViewBag.Users = JsonConvert.DeserializeObject<IEnumerable<User>>(content);

Console.WriteLine($"This is {url} Invoke");

return View();

1. }