1.为什么要用thrift js C#？

    1.1 首先，js 通过 thrift 访问C#，实际上是一种c/s模式。thrift是通信工具，js是客户端，C#是服务端。

    1.2 使用js直接与thrift server通信。让web开发变得更简单。如果使用Web Service，你需要自己去实现C/S两端的序列化与反序列化操作，还需要自行处理异常，降低了开发效率。而thrift则会自动生成两端的操作类，你 只需要处理方法内部的逻辑即可。

    1.3 js直接与thrift server通信，可以提高性能，并且用C#开发server，开发效率也非常高。（那些web service弱爆了）

    1.4 当然，我们不能只看到优点。用这种方法也有比较明显的缺点：如果使用web service，则网页与web service可以封装在一个工程里，部署到IIS上后，可以存在于一个网站内。而使用了thrift后，你还需要手动管理thrift程序。包括：

        1.4.1 你需要拥有服务端的绝对控制权，比如，你可以直接登录服务器的操作系统上进行操作。因此，如果你只有一个网页空间，则不适合这种方法。当然，你也可以用 web service里绑定thrift，但这样你又需要自己手动进行序列化与反序列操作，而且两次转换让性能更低，有违初衷

        1.4.2 给thrift server程序增加自动启动，与监视程序，来完成thrift的崩溃后自动重启。

2.环境

    Win7 - VS2012 - .net 4.0 C# 控制台工程（用来承载thrift）

    Win7 - VS2012 - .net 4.0 C# Web工程（用来调试js，超方便）

3.步骤（以下步骤，对于小白来说，有些困难。QQ讨论群：23152359 ）

    3.1 去thrift官方下载thrift库，目前是0.9.0。

    3.2 去thrift官方下载编译好的win下的thrift编译器，是一个exe文件。

    3.3 写一个数据结构定义文件。我在这里只是用了服务，没有定义自定义数据结构。

        data.txt：

1 service UserStorage2 {3     i32 Sum( 1: i32 arg\_number1, 2: i32 arg\_number2),4 string GetString()5 }

    3.4 命令行下，用thrift编译器，对它进行编译：  
        run.bat：

1 thrift-0.9.0.exe --gen csharp data.txt2 thrift-0.9.0.exe --gen js data.txt3 pause

    3.5 建立一个名字为CSharpServer的C#控制台工程，.net 4.0的。

    3.6 为这个工程，添加现有项目：thrift库目录\thrift-0.9.0\lib\csharp\src\Thrift.csproj，然后引用这个项目。

    3.7 把thrift编译出来的UserStorage.cs（在gen-csharp目录里），拖动到解决方案管理器里的CSharpServer项目的根目录下，UserStorage.cs与Program.cs应该在同一级。

    3.8 在CSharpServer项目里创建一个UserStorage的处理类UserStorageHandle.cs：（它应该与UserStorage.cs与Program.cs在同一级）

[复制代码](javascript:void(0);)

1 using System; 2 using System.Collections.Generic; 3 using System.Linq; 4 using System.Text; 5 6namespace CSharpServer 7 { 8 public class UserStorageHandle : UserStorage.Iface 9     {10 11 publicUserStorageHandle()12         {13             ;14         }15 16 public int Sum(int arg\_number1, intarg\_number2)17         {18 int result = arg\_number1 + arg\_number2;19             Console.WriteLine(DateTime.Now.ToString() + " 收到请求：Sum，参数：arg\_number1 = " + arg\_number1.ToString() + "，arg\_number2 = " + arg\_number2.ToString() + "，返回：result = " + result.ToString());20 return result;21         }22 23 private static int Counter = 0;24 25 public stringGetString()26         {27 int currentCounter = System.Threading.Interlocked.Increment(refUserStorageHandle.Counter);28             Console.WriteLine(DateTime.Now.ToString() + " 收到请求：GetString，参数：没有，返回：result = \"thrift is OK : " + currentCounter.ToString() + "\"");29 return"thrift is OK : " + currentCounter.ToString();30         }31 32     }33 }

[复制代码](javascript:void(0);)

    3.9 主程序Program.cs：

[复制代码](javascript:void(0);)

1 using System; 2 using System.Collections.Generic; 3 using System.Linq; 4 using System.Text; 5using Thrift.Transport; 6 using Thrift.Protocol; 7 using Thrift.Server; 8 using System.Net; 9 10namespace CSharpServer11 {12 class Program13     {14 private static HttpListener httpListener =null;15 private static THttpHandler httpServer = null;16 17 static void Main(string[] args)18         {19 stringserviceUrl = "http://localhost:99/";20 try21             {22                 UserStorageHandle handle = newUserStorageHandle();23                 UserStorage.Processor processor = newUserStorage.Processor(handle);24                 TProtocolFactory protocolFactory = newTJSONProtocol.Factory();25 26                 Program.httpServer = new THttpHandler(processor, protocolFactory);27 28                 Program.httpListener = new HttpListener();29                 Program.httpListener.Prefixes.Add(serviceUrl);30                 Program.httpListener.Start();31                 IAsyncResult result = Program.httpListener.BeginGetContext(newAsyncCallback(WebRequestCallback), null);32                 Console.WriteLine("Thrift 服务成功启动，网址是 " + serviceUrl);33             }34 catch (System.Exception ex)35             {36                 Console.WriteLine("发生错误：" + ex.Message);37                 Console.WriteLine("按回车键退出");38                Console.ReadLine();39 return;40             }41             Console.WriteLine("若需结束程序，请直接关闭窗口，或按回车。");42             Console.ReadLine();43         }44 45 public static voidWebRequestCallback(IAsyncResult result)46         {47 if (Program.httpListener == null)48             {49                 Console.WriteLine("发生错误：HttpListener已经被关闭");50                 Console.WriteLine("按回车键退出");51                 Console.ReadLine();52 return;53             }54 55             HttpListenerContext httpListenerContext = Program.httpListener.EndGetContext(result);56 57             Program.httpListener.BeginGetContext(new AsyncCallback(WebRequestCallback), null);58 59             Program.httpServer.ProcessRequest(httpListenerContext);60         }61 62     }63 }

[复制代码](javascript:void(0);)

    3.10 按F5，启动CSharpServer项目。

    3.11 新开一个VS2012（内存不够的赶快去买），建立一个叫JsProject的C# .net 4.0 Web空工程。

    3.12 去jquery官网下载jquery-1.9.1.js（mini版也行，随便）

    3.13 把thrift编译的js文件data\_types.js与UserStorage.js，thrift库的js库文件（thrift库目 录\thrift-0.9.0\lib\js\thrift.js），以及刚下载的jq文件jquery-1.9.1.js，全部拖动到解决方案资源管理 器的JsProject项目的根目录下：

        data\_types.js \ jquery-1.9.1.js \ thrift.js \ UserStorage.js 应该与Web.config在同一级。

    3.14 在根目录下创建一个Test.html文件，Test.html 应该与Web.config在同一级：

            Test.html：

[复制代码](javascript:void(0);)

1 <!DOCTYPE html> 2 <html xmlns="http://www.w3.org/1999/xhtml"> 3 <head> 4 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/> 5     <title></title> 6     <script type="text/javascript" src="jquery-1.9.1.js"></script> 7     <script type="text/javascript"src="data\_types.js"></script> 8     <script type="text/javascript" src="thrift.js"></script> 9     <script type="text/javascript" src="UserStorage.js"></script>10 11     <script>12         $(document).ready(function ()13         {14 var debugPosation = 0;15 try16             {17 var transport =new Thrift.Transport("http://localhost:99");18 var protocol = new Thrift.Protocol(transport);19 var client =new UserStorageClient(protocol);20 21 var result\_GetString = client.GetString();22 var result\_Sum = client.Sum(255, 322);23             }24 catch (e)25             {26                 alert("出错鸟：" + e.message);27            }28         });29     </script>30 </head>31 <body>32 </body>33 </html>

[复制代码](javascript:void(0);)

    3.15 给Test.html的 “ var debugPosation = 0; ” 这一行，下断点，然后F5，就可以看到效果了。

4.注意两个问题

    4.1 由于web服务一般都是80端口，如果thrift端口与其不同，js在访问时会遇到跨域问题，js报错e.message = "拒绝访问"。但是，对于传说中的windows server，完全没有问题，因为，windows server支持.net的传说中的端口复用功能！因此，假设部署后，网站的iis的url为：<http://www.baidu.com/XXX>，则你最好给thrift server留一个二级域名，比如<http://www.baidu.com/ThriftService> ，把这个留给thrift。最后，直接把thrift server的监听url，设置为这个 <http://www.baidu.com/ThriftService> 就OK了，不会有端口冲突问题，因为win Server + .net 支持tcp端口复用！！！！ 如果你的网站的url还带有端口，比如 <http://www.baidu.com:9999/XXX>，或者IP形式：<http://11.2.33.44:9999/XXX>，则thriftservice也需要与它的端口相同，比如：<http://www.baidu:9999/ThriftService> 或 <http://11.2.33.44:9999/ThriftService> 。问，网站端口与thriftserver端口并不是80，而是9999，有没有问题？答：木有问题，端口复用功能依然会发挥作用。不过，端口复用技术， 是通过让IIS和ThriftServer，把端口监听的工作，都交给System.exe去做，所有如果要做防火墙，要注意一下。防火墙参见这个帖子：<http://www.cnblogs.com/xxxteam/archive/2013/04/22/3036082.html>

    4.2 网址的最后一个斜杠问题。js访问thrift service，网址最后一个字符，是不带斜杠的。比如<http://www.baidu.com/XXX/>，这个就是错误的，因为最后一个是斜杠。应该去掉最后一个斜杠，更正为： <http://www.baidu.com/XXX>  。而ThriftServer的监听网址，则最后一个斜杠是不能省去的。坑爹吧？当然了，这个问题可能会因为日后的补丁被修正，所以大家自己留意一下。

5.为什么不能使用js + thrift server方案？

    看到这里傻眼了吧？上面白忙活大半天，但最后居然发现这个方案有问题？是的。

    5.1 Thrift的定义，仅仅是一个通信工具，它不是能7\*24运行的类似于iis这种服务提供程序。相对于wcf来说，thrift简单，所以高效。但正是 因为它简单，所以它缺少很多在复杂网络环境下充当服务端的功能。在实际使用中，如果把它当成webService Server，则经常出问题。而且很多问题是try catch也抓不住，直接导致宿主程序崩溃的。所以，如果希望构建一个稳定的webService server，还是使用wcf之类的稳定方案为妙。

        不过，这并不是说thrift就不行。thrift可以在系统内部来进行数据传输，但绝对不能放在外部。

    5.2 js + thrift不支持异步调用！原因是，js只有以ajax方式进行远程服务调用才具有异步功能，但thrift封装的远程服务调用，不是ajax方式的。

   其他更多问题详见 <http://www.cnblogs.com/xxxteam/archive/2013/04/24/3040839.html>

来源： <<http://www.cnblogs.com/xxxteam/archive/2013/04/15/3023159.html>>