[**.net 调用webservice 总结**](http://www.cnblogs.com/eagle1986/archive/2012/09/03/2669699.html)

wsdl是在调用[web服务](https://www.baidu.com/s?wd=web%E6%9C%8D%E5%8A%A1&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dBn1b3ryNhryDYmW9hmhm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnH6zrH64nWT4PjcLPWckrHnsn0)时，由服务器发布的调用规范描述，这个描述是[XML格式](https://www.baidu.com/s?wd=XML%E6%A0%BC%E5%BC%8F&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dBn1b3ryNhryDYmW9hmhm40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnH6zrH64nWT4PjcLPWckrHnsn0)的。

网络服务描述语言是Web Service的描述语言，它包含一系列描述某个web service的定义。

最近做一个项目，由于是在别人框架里开发app,导致了很多限制，其中一个就是不能直接引用webservice 。

我们都知道，调用webserivice 最简单的方法就是在 "引用"  那里点击右键，然后选择"引用web服务"，再输入服务地址。

确定后，会生成一个app.config 里面就会自动生成了一些配置信息。

现在正在做的这个项目就不能这么干。后来经过一番搜索，就找出另外几种动态调用webservice 的方法。

废话少说，下面是webservice 代码

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Services;

namespace TestWebService

{

/// <summary>

/// Service1 的摘要说明

/// </summary>

[WebService(Namespace = "http://tempuri.org/",Description="我的Web服务")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

[System.ComponentModel.ToolboxItem(false)]

// 若要允许使用 ASP.NET AJAX 从脚本中调用此 Web 服务，请取消对下行的注释。

// [System.Web.Script.Services.ScriptService]

public class TestWebService : System.Web.Services.WebService

{

[WebMethod]

public string HelloWorld()

{

return "测试Hello World";

}

[WebMethod]

public string Test()

{

return "测试Test";

}

[WebMethod(CacheDuration = 60,Description = "测试")]

public List<String> GetPersons()

{

List<String> list = new List<string>();

list.Add("测试一");

list.Add("测试二");

list.Add("测试三");

return list;

}

}

}

[复制代码](javascript:void(0);)

动态调用示例：

方法一：

看到很多动态调用WebService都只是动态调用地址而已，下面发一个不光是根据地址调用，方法名也可以自己指定的,主要原理是根据指定的WebService地址的WSDL,然后解析模拟生成一个代理类,通过反射调用里面的方法

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

using System;

using System.IO;

using System.Collections.Generic;

using System.Linq;

using System.Collections;

using System.Web;

using System.Net;

using System.Reflection;

using System.CodeDom;

using System.CodeDom.Compiler;

using System.Web.Services;

using System.Text;

using System.Web.Services.Description;

using System.Web.Services.Protocols;

using System.Xml.Serialization;

using System.Windows.Forms;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

WebClient client = new WebClient();

String url = "http://localhost:3182/Service1.asmx?WSDL";//这个地址可以写在Config文件里面，这里取出来就行了.在原地址后面加上： ?WSDL

Stream stream = client.OpenRead(url);

ServiceDescription description = ServiceDescription.Read(stream);

ServiceDescriptionImporter importer = new ServiceDescriptionImporter();//创建客户端代理代理类。

importer.ProtocolName = "Soap"; //指定访问协议。

importer.Style = ServiceDescriptionImportStyle.Client; //生成客户端代理。

importer.CodeGenerationOptions = CodeGenerationOptions.GenerateProperties | CodeGenerationOptions.GenerateNewAsync;

importer.AddServiceDescription(description, null, null); //添加WSDL文档。

CodeNamespace nmspace = new CodeNamespace(); //命名空间

nmspace.Name = "TestWebService";

CodeCompileUnit unit = new CodeCompileUnit();

unit.Namespaces.Add(nmspace);

ServiceDescriptionImportWarnings warning = importer.Import(nmspace, unit);

CodeDomProvider provider = CodeDomProvider.CreateProvider("CSharp");

CompilerParameters parameter = new CompilerParameters();

parameter.GenerateExecutable = false;

parameter.OutputAssembly = "MyTest.dll";//输出程序集的名称

parameter.ReferencedAssemblies.Add("System.dll");

parameter.ReferencedAssemblies.Add("System.XML.dll");

parameter.ReferencedAssemblies.Add("System.Web.Services.dll");

parameter.ReferencedAssemblies.Add("System.Data.dll");

CompilerResults result = provider.CompileAssemblyFromDom(parameter, unit);

if (result.Errors.HasErrors)

{

// 显示编译错误信息

}

Assembly asm = Assembly.LoadFrom("MyTest.dll");//加载前面生成的程序集

Type t = asm.GetType("TestWebService.TestWebService");

object o = Activator.CreateInstance(t);

MethodInfo method = t.GetMethod("GetPersons");//GetPersons是服务端的方法名称,你想调用服务端的什么方法都可以在这里改,最好封装一下

String[] item = (String[])method.Invoke(o, null);

//注：method.Invoke(o, null)返回的是一个Object,如果你服务端返回的是DataSet,这里也是用(DataSet)method.Invoke(o, null)转一下就行了,method.Invoke(0,null)这里的null可以传调用方法需要的参数,string[]形式的

foreach (string str in item)

Console.WriteLine(str);

//上面是根据WebService地址，模似生成一个代理类,如果你想看看生成的代码文件是什么样子，可以用以下代码保存下来，默认是保存在bin目录下面

TextWriter writer = File.CreateText("MyTest.cs");

provider.GenerateCodeFromCompileUnit(unit, writer, null);

writer.Flush();

writer.Close();

}

}

}

[复制代码](javascript:void(0);)

在网上找了一个更为详细的

<http://blog.csdn.net/ysq5202121/article/details/6942813>

方法二：利用 wsdl.exe生成webservice代理类:

根据提供的wsdl生成webservice代理类，然后在代码里引用这个类文件。

步骤：1、在开始菜单找到  Microsoft Visual Studio 2010 下面的Visual Studio Tools ， 点击Visual Studio 命令提示(2010)，打开命令行。

          2、 在命令行中输入：  wsdl /language:c# /n:TestDemo /out:d:/Temp/TestService.cs <http://jm1.services.gmcc.net/ad/Services/AD.asmx?wsdl>

                这句命令行的意思是：对最后面的服务地址进行编译，在D盘temp 目录下生成testservice文件。

          3、 把上面命令编译后的cs文件，复制到我们项目中，在项目代码中可以直接new 一个出来后，可以进行调用。

   贴出由命令行编译出来的代码：

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

//------------------------------------------------------------------------------

// <auto-generated>

// 此代码由工具生成。

// 运行时版本:4.0.30319.225

//

// 对此文件的更改可能会导致不正确的行为，并且如果

// 重新生成代码，这些更改将会丢失。

// </auto-generated>

//------------------------------------------------------------------------------

//

// 此源代码由 wsdl 自动生成, Version=4.0.30319.1。

//

namespace Bingosoft.Module.SurveyQuestionnaire.DAL {

using System;

using System.Diagnostics;

using System.Xml.Serialization;

using System.ComponentModel;

using System.Web.Services.Protocols;

using System.Web.Services;

using System.Data;

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

[System.Web.Services.WebServiceBindingAttribute(Name="WebserviceForILookSoap", Namespace="http://tempuri.org/")]

public partial class WebserviceForILook : System.Web.Services.Protocols.SoapHttpClientProtocol {

private System.Threading.SendOrPostCallback GetRecordNumOperationCompleted;

private System.Threading.SendOrPostCallback GetVoteListOperationCompleted;

private System.Threading.SendOrPostCallback VoteOperationCompleted;

private System.Threading.SendOrPostCallback GiveUpOperationCompleted;

private System.Threading.SendOrPostCallback GetQuestionTaskListOperationCompleted;

/// <remarks/>

public WebserviceForILook() {

this.Url = "http://st1.services.gmcc.net/qnaire/Services/WebserviceForILook.asmx";

}

/// <remarks/>

public event GetRecordNumCompletedEventHandler GetRecordNumCompleted;

/// <remarks/>

public event GetVoteListCompletedEventHandler GetVoteListCompleted;

/// <remarks/>

public event VoteCompletedEventHandler VoteCompleted;

/// <remarks/>

public event GiveUpCompletedEventHandler GiveUpCompleted;

/// <remarks/>

public event GetQuestionTaskListCompletedEventHandler GetQuestionTaskListCompleted;

/// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/GetRecordNum", RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/", Use=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]

public int[] GetRecordNum(string appcode, string userID) {

object[] results = this.Invoke("GetRecordNum", new object[] {

appcode,

userID});

return ((int[])(results[0]));

}

/// <remarks/>

public System.IAsyncResult BeginGetRecordNum(string appcode, string userID, System.AsyncCallback callback, object asyncState) {

return this.BeginInvoke("GetRecordNum", new object[] {

appcode,

userID}, callback, asyncState);

}

/// <remarks/>

public int[] EndGetRecordNum(System.IAsyncResult asyncResult) {

object[] results = this.EndInvoke(asyncResult);

return ((int[])(results[0]));

}

/// <remarks/>

public void GetRecordNumAsync(string appcode, string userID) {

this.GetRecordNumAsync(appcode, userID, null);

}

/// <remarks/>

public void GetRecordNumAsync(string appcode, string userID, object userState) {

if ((this.GetRecordNumOperationCompleted == null)) {

this.GetRecordNumOperationCompleted = new System.Threading.SendOrPostCallback(this.OnGetRecordNumOperationCompleted);

}

this.InvokeAsync("GetRecordNum", new object[] {

appcode,

userID}, this.GetRecordNumOperationCompleted, userState);

}

private void OnGetRecordNumOperationCompleted(object arg) {

if ((this.GetRecordNumCompleted != null)) {

System.Web.Services.Protocols.InvokeCompletedEventArgs invokeArgs = ((System.Web.Services.Protocols.InvokeCompletedEventArgs)(arg));

this.GetRecordNumCompleted(this, new GetRecordNumCompletedEventArgs(invokeArgs.Results, invokeArgs.Error, invokeArgs.Cancelled, invokeArgs.UserState));

}

}

/// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/GetVoteList", RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/", Use=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]

public System.Data.DataSet GetVoteList(string appcode, string userID) {

object[] results = this.Invoke("GetVoteList", new object[] {

appcode,

userID});

return ((System.Data.DataSet)(results[0]));

}

/// <remarks/>

public System.IAsyncResult BeginGetVoteList(string appcode, string userID, System.AsyncCallback callback, object asyncState) {

return this.BeginInvoke("GetVoteList", new object[] {

appcode,

userID}, callback, asyncState);

}

/// <remarks/>

public System.Data.DataSet EndGetVoteList(System.IAsyncResult asyncResult) {

object[] results = this.EndInvoke(asyncResult);

return ((System.Data.DataSet)(results[0]));

}

/// <remarks/>

public void GetVoteListAsync(string appcode, string userID) {

this.GetVoteListAsync(appcode, userID, null);

}

/// <remarks/>

public void GetVoteListAsync(string appcode, string userID, object userState) {

if ((this.GetVoteListOperationCompleted == null)) {

this.GetVoteListOperationCompleted = new System.Threading.SendOrPostCallback(this.OnGetVoteListOperationCompleted);

}

this.InvokeAsync("GetVoteList", new object[] {

appcode,

userID}, this.GetVoteListOperationCompleted, userState);

}

private void OnGetVoteListOperationCompleted(object arg) {

if ((this.GetVoteListCompleted != null)) {

System.Web.Services.Protocols.InvokeCompletedEventArgs invokeArgs = ((System.Web.Services.Protocols.InvokeCompletedEventArgs)(arg));

this.GetVoteListCompleted(this, new GetVoteListCompletedEventArgs(invokeArgs.Results, invokeArgs.Error, invokeArgs.Cancelled, invokeArgs.UserState));

}

}

/// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/Vote", RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/", Use=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]

public bool Vote(string appcode, string userID, string qTaskID, string answer) {

object[] results = this.Invoke("Vote", new object[] {

appcode,

userID,

qTaskID,

answer});

return ((bool)(results[0]));

}

/// <remarks/>

public System.IAsyncResult BeginVote(string appcode, string userID, string qTaskID, string answer, System.AsyncCallback callback, object asyncState) {

return this.BeginInvoke("Vote", new object[] {

appcode,

userID,

qTaskID,

answer}, callback, asyncState);

}

/// <remarks/>

public bool EndVote(System.IAsyncResult asyncResult) {

object[] results = this.EndInvoke(asyncResult);

return ((bool)(results[0]));

}

/// <remarks/>

public void VoteAsync(string appcode, string userID, string qTaskID, string answer) {

this.VoteAsync(appcode, userID, qTaskID, answer, null);

}

/// <remarks/>

public void VoteAsync(string appcode, string userID, string qTaskID, string answer, object userState) {

if ((this.VoteOperationCompleted == null)) {

this.VoteOperationCompleted = new System.Threading.SendOrPostCallback(this.OnVoteOperationCompleted);

}

this.InvokeAsync("Vote", new object[] {

appcode,

userID,

qTaskID,

answer}, this.VoteOperationCompleted, userState);

}

private void OnVoteOperationCompleted(object arg) {

if ((this.VoteCompleted != null)) {

System.Web.Services.Protocols.InvokeCompletedEventArgs invokeArgs = ((System.Web.Services.Protocols.InvokeCompletedEventArgs)(arg));

this.VoteCompleted(this, new VoteCompletedEventArgs(invokeArgs.Results, invokeArgs.Error, invokeArgs.Cancelled, invokeArgs.UserState));

}

}

/// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/GiveUp", RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/", Use=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]

public bool GiveUp(string appcode, string userID, string qTaskID) {

object[] results = this.Invoke("GiveUp", new object[] {

appcode,

userID,

qTaskID});

return ((bool)(results[0]));

}

/// <remarks/>

public System.IAsyncResult BeginGiveUp(string appcode, string userID, string qTaskID, System.AsyncCallback callback, object asyncState) {

return this.BeginInvoke("GiveUp", new object[] {

appcode,

userID,

qTaskID}, callback, asyncState);

}

/// <remarks/>

public bool EndGiveUp(System.IAsyncResult asyncResult) {

object[] results = this.EndInvoke(asyncResult);

return ((bool)(results[0]));

}

/// <remarks/>

public void GiveUpAsync(string appcode, string userID, string qTaskID) {

this.GiveUpAsync(appcode, userID, qTaskID, null);

}

/// <remarks/>

public void GiveUpAsync(string appcode, string userID, string qTaskID, object userState) {

if ((this.GiveUpOperationCompleted == null)) {

this.GiveUpOperationCompleted = new System.Threading.SendOrPostCallback(this.OnGiveUpOperationCompleted);

}

this.InvokeAsync("GiveUp", new object[] {

appcode,

userID,

qTaskID}, this.GiveUpOperationCompleted, userState);

}

private void OnGiveUpOperationCompleted(object arg) {

if ((this.GiveUpCompleted != null)) {

System.Web.Services.Protocols.InvokeCompletedEventArgs invokeArgs = ((System.Web.Services.Protocols.InvokeCompletedEventArgs)(arg));

this.GiveUpCompleted(this, new GiveUpCompletedEventArgs(invokeArgs.Results, invokeArgs.Error, invokeArgs.Cancelled, invokeArgs.UserState));

}

}

/// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/GetQuestionTaskList", RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/", Use=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]

public System.Data.DataSet GetQuestionTaskList(string appcode, string userID) {

object[] results = this.Invoke("GetQuestionTaskList", new object[] {

appcode,

userID});

return ((System.Data.DataSet)(results[0]));

}

/// <remarks/>

public System.IAsyncResult BeginGetQuestionTaskList(string appcode, string userID, System.AsyncCallback callback, object asyncState) {

return this.BeginInvoke("GetQuestionTaskList", new object[] {

appcode,

userID}, callback, asyncState);

}

/// <remarks/>

public System.Data.DataSet EndGetQuestionTaskList(System.IAsyncResult asyncResult) {

object[] results = this.EndInvoke(asyncResult);

return ((System.Data.DataSet)(results[0]));

}

/// <remarks/>

public void GetQuestionTaskListAsync(string appcode, string userID) {

this.GetQuestionTaskListAsync(appcode, userID, null);

}

/// <remarks/>

public void GetQuestionTaskListAsync(string appcode, string userID, object userState) {

if ((this.GetQuestionTaskListOperationCompleted == null)) {

this.GetQuestionTaskListOperationCompleted = new System.Threading.SendOrPostCallback(this.OnGetQuestionTaskListOperationCompleted);

}

this.InvokeAsync("GetQuestionTaskList", new object[] {

appcode,

userID}, this.GetQuestionTaskListOperationCompleted, userState);

}

private void OnGetQuestionTaskListOperationCompleted(object arg) {

if ((this.GetQuestionTaskListCompleted != null)) {

System.Web.Services.Protocols.InvokeCompletedEventArgs invokeArgs = ((System.Web.Services.Protocols.InvokeCompletedEventArgs)(arg));

this.GetQuestionTaskListCompleted(this, new GetQuestionTaskListCompletedEventArgs(invokeArgs.Results, invokeArgs.Error, invokeArgs.Cancelled, invokeArgs.UserState));

}

}

/// <remarks/>

public new void CancelAsync(object userState) {

base.CancelAsync(userState);

}

}

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

public delegate void GetRecordNumCompletedEventHandler(object sender, GetRecordNumCompletedEventArgs e);

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

public partial class GetRecordNumCompletedEventArgs : System.ComponentModel.AsyncCompletedEventArgs {

private object[] results;

internal GetRecordNumCompletedEventArgs(object[] results, System.Exception exception, bool cancelled, object userState) :

base(exception, cancelled, userState) {

this.results = results;

}

/// <remarks/>

public int[] Result {

get {

this.RaiseExceptionIfNecessary();

return ((int[])(this.results[0]));

}

}

}

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

public delegate void GetVoteListCompletedEventHandler(object sender, GetVoteListCompletedEventArgs e);

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

public partial class GetVoteListCompletedEventArgs : System.ComponentModel.AsyncCompletedEventArgs {

private object[] results;

internal GetVoteListCompletedEventArgs(object[] results, System.Exception exception, bool cancelled, object userState) :

base(exception, cancelled, userState) {

this.results = results;

}

/// <remarks/>

public System.Data.DataSet Result {

get {

this.RaiseExceptionIfNecessary();

return ((System.Data.DataSet)(this.results[0]));

}

}

}

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

public delegate void VoteCompletedEventHandler(object sender, VoteCompletedEventArgs e);

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

public partial class VoteCompletedEventArgs : System.ComponentModel.AsyncCompletedEventArgs {

private object[] results;

internal VoteCompletedEventArgs(object[] results, System.Exception exception, bool cancelled, object userState) :

base(exception, cancelled, userState) {

this.results = results;

}

/// <remarks/>

public bool Result {

get {

this.RaiseExceptionIfNecessary();

return ((bool)(this.results[0]));

}

}

}

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

public delegate void GiveUpCompletedEventHandler(object sender, GiveUpCompletedEventArgs e);

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

public partial class GiveUpCompletedEventArgs : System.ComponentModel.AsyncCompletedEventArgs {

private object[] results;

internal GiveUpCompletedEventArgs(object[] results, System.Exception exception, bool cancelled, object userState) :

base(exception, cancelled, userState) {

this.results = results;

}

/// <remarks/>

public bool Result {

get {

this.RaiseExceptionIfNecessary();

return ((bool)(this.results[0]));

}

}

}

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

public delegate void GetQuestionTaskListCompletedEventHandler(object sender, GetQuestionTaskListCompletedEventArgs e);

/// <remarks/>

[System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "4.0.30319.1")]

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.ComponentModel.DesignerCategoryAttribute("code")]

public partial class GetQuestionTaskListCompletedEventArgs : System.ComponentModel.AsyncCompletedEventArgs {

private object[] results;

internal GetQuestionTaskListCompletedEventArgs(object[] results, System.Exception exception, bool cancelled, object userState) :

base(exception, cancelled, userState) {

this.results = results;

}

/// <remarks/>

public System.Data.DataSet Result {

get {

this.RaiseExceptionIfNecessary();

return ((System.Data.DataSet)(this.results[0]));

}

}

}

}

[复制代码](javascript:void(0);)

更为详细的可以参见：<http://blog.csdn.net/slimboy123/article/details/4344914>

方法三：利用http 协议的get  和post

这是最为灵活的方法。

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

using System;

using System.Collections;

using System.IO;

using System.Net;

using System.Text;

using System.Xml;

using System.Xml.Serialization;

namespace Bingosoft.RIA.Common

{

/// <summary>

/// 利用WebRequest/WebResponse进行WebService调用的类

/// </summary>

public class WebServiceCaller

{

#region Tip:使用说明

//webServices 应该支持Get和Post调用，在web.config应该增加以下代码

//<webServices>

// <protocols>

// <add name="HttpGet"/>

// <add name="HttpPost"/>

// </protocols>

//</webServices>

//调用示例：

//Hashtable ht = new Hashtable(); //Hashtable 为webservice所需要的参数集

//ht.Add("str", "test");

//ht.Add("b", "true");

//XmlDocument xx = WebSvcCaller.QuerySoapWebService("http://localhost:81/service.asmx", "HelloWorld", ht);

//MessageBox.Show(xx.OuterXml);

#endregion

/// <summary>

/// 需要WebService支持Post调用

/// </summary>

public static XmlDocument QueryPostWebService(String URL, String MethodName, Hashtable Pars)

{

HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(URL + "/" + MethodName);

request.Method = "POST";

request.ContentType = "application/x-www-form-urlencoded";

SetWebRequest(request);

byte[] data = EncodePars(Pars);

WriteRequestData(request, data);

return ReadXmlResponse(request.GetResponse());

}

/// <summary>

/// 需要WebService支持Get调用

/// </summary>

public static XmlDocument QueryGetWebService(String URL, String MethodName, Hashtable Pars)

{

HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(URL + "/" + MethodName + "?" + ParsToString(Pars));

request.Method = "GET";

request.ContentType = "application/x-www-form-urlencoded";

SetWebRequest(request);

return ReadXmlResponse(request.GetResponse());

}

/// <summary>

/// 通用WebService调用(Soap),参数Pars为String类型的参数名、参数值

/// </summary>

public static XmlDocument QuerySoapWebService(String URL, String MethodName, Hashtable Pars)

{

if (\_xmlNamespaces.ContainsKey(URL))

{

return QuerySoapWebService(URL, MethodName, Pars, \_xmlNamespaces[URL].ToString());

}

else

{

return QuerySoapWebService(URL, MethodName, Pars, GetNamespace(URL));

}

}

private static XmlDocument QuerySoapWebService(String URL, String MethodName, Hashtable Pars, string XmlNs)

{

\_xmlNamespaces[URL] = XmlNs;//加入缓存，提高效率

HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(URL);

request.Method = "POST";

request.ContentType = "text/xml; charset=utf-8";

request.Headers.Add("SOAPAction", "\"" + XmlNs + (XmlNs.EndsWith("/") ? "" : "/") + MethodName + "\"");

SetWebRequest(request);

byte[] data = EncodeParsToSoap(Pars, XmlNs, MethodName);

WriteRequestData(request, data);

XmlDocument doc = new XmlDocument(), doc2 = new XmlDocument();

doc = ReadXmlResponse(request.GetResponse());

XmlNamespaceManager mgr = new XmlNamespaceManager(doc.NameTable);

mgr.AddNamespace("soap", "http://schemas.xmlsoap.org/soap/envelope/");

String RetXml = doc.SelectSingleNode("//soap:Body/\*/\*", mgr).InnerXml;

doc2.LoadXml("<root>" + RetXml + "</root>");

AddDelaration(doc2);

return doc2;

}

private static string GetNamespace(String URL)

{

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(URL + "?WSDL");

SetWebRequest(request);

WebResponse response = request.GetResponse();

StreamReader sr = new StreamReader(response.GetResponseStream(), Encoding.UTF8);

XmlDocument doc = new XmlDocument();

doc.LoadXml(sr.ReadToEnd());

sr.Close();

return doc.SelectSingleNode("//@targetNamespace").Value;

}

private static byte[] EncodeParsToSoap(Hashtable Pars, String XmlNs, String MethodName)

{

XmlDocument doc = new XmlDocument();

doc.LoadXml("<soap:Envelope xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\" xmlns:soap=\"http://schemas.xmlsoap.org/soap/envelope/\"></soap:Envelope>");

AddDelaration(doc);

//XmlElement soapBody = doc.createElement\_x\_x("soap", "Body", "http://schemas.xmlsoap.org/soap/envelope/");

XmlElement soapBody = doc.CreateElement("soap", "Body", "http://schemas.xmlsoap.org/soap/envelope/");

//XmlElement soapMethod = doc.createElement\_x\_x(MethodName);

XmlElement soapMethod = doc.CreateElement(MethodName);

soapMethod.SetAttribute("xmlns", XmlNs);

foreach (string k in Pars.Keys)

{

//XmlElement soapPar = doc.createElement\_x\_x(k);

XmlElement soapPar = doc.CreateElement(k);

soapPar.InnerXml = ObjectToSoapXml(Pars[k]);

soapMethod.AppendChild(soapPar);

}

soapBody.AppendChild(soapMethod);

doc.DocumentElement.AppendChild(soapBody);

return Encoding.UTF8.GetBytes(doc.OuterXml);

}

private static string ObjectToSoapXml(object o)

{

XmlSerializer mySerializer = new XmlSerializer(o.GetType());

MemoryStream ms = new MemoryStream();

mySerializer.Serialize(ms, o);

XmlDocument doc = new XmlDocument();

doc.LoadXml(Encoding.UTF8.GetString(ms.ToArray()));

if (doc.DocumentElement != null)

{

return doc.DocumentElement.InnerXml;

}

else

{

return o.ToString();

}

}

/// <summary>

/// 设置凭证与超时时间

/// </summary>

/// <param name="request"></param>

private static void SetWebRequest(HttpWebRequest request)

{

request.Credentials = CredentialCache.DefaultCredentials;

request.Timeout = 10000;

}

private static void WriteRequestData(HttpWebRequest request, byte[] data)

{

request.ContentLength = data.Length;

Stream writer = request.GetRequestStream();

writer.Write(data, 0, data.Length);

writer.Close();

}

private static byte[] EncodePars(Hashtable Pars)

{

return Encoding.UTF8.GetBytes(ParsToString(Pars));

}

private static String ParsToString(Hashtable Pars)

{

StringBuilder sb = new StringBuilder();

foreach (string k in Pars.Keys)

{

if (sb.Length > 0)

{

sb.Append("&");

}

//sb.Append(HttpUtility.UrlEncode(k) + "=" + HttpUtility.UrlEncode(Pars[k].ToString()));

}

return sb.ToString();

}

private static XmlDocument ReadXmlResponse(WebResponse response)

{

StreamReader sr = new StreamReader(response.GetResponseStream(), Encoding.UTF8);

String retXml = sr.ReadToEnd();

sr.Close();

XmlDocument doc = new XmlDocument();

doc.LoadXml(retXml);

return doc;

}

private static void AddDelaration(XmlDocument doc)

{

XmlDeclaration decl = doc.CreateXmlDeclaration("1.0", "utf-8", null);

doc.InsertBefore(decl, doc.DocumentElement);

}

private static Hashtable \_xmlNamespaces = new Hashtable();//缓存xmlNamespace，避免重复调用GetNamespace

}

}