HTTP GET 及 HTTP POST 預設為停用

# ajax

cache:false, //防止ie8一直取到舊資料的話，請設定為false

async: false,//最後要return或是等等需要用到result的value的話，必須設定為false(就會變成同步執行),不然就不設定，預設是true

data: "{}",//不用傳參數的話，放個大括弧就好

我們通常都會透過ScriptManager來註冊Service Reference，

就可以直接呼叫Wcf中的function了，

但今天如果不是使用Asp.Net的話，

我們也可以透過JQuery，來對Wcf進行呼叫。

# Aspx

|  |  |  |  |
| --- | --- | --- | --- |
|  | **單一字串** | **object** | **List<object>** |
| Server丟出來的 | hihi | "{\"a\":123,\"b\":\"333\"}" | "[{\"a\":123,\"b\":\"333\"},{\"a\":123,\"b\":\"444\"}]" |
| $.ajax({  url: "AppCode/json.aspx",  type: 'POST', |  |  |  |
| $.ajax({  url: "AppCode/json.aspx",  dataType: 'json',  type: 'POST', | error |  |  |

# Asmx

## Asmx 輸出string

|  |  |  |
| --- | --- | --- |
| **Asmx 輸出string** | | |
|  | **<string>xxx</string>** | **d:xxx**  **contentType: "application/json; charset=utf-8",** |
| **String** |  |  |
| **object** |  |  |
| **List<object>** |  |  |
| **總結** | 就是用<string>包起來,就是放在d:後面,只是外包裝不一樣而已 | |

## Asmx 輸出object

|  |  |  |
| --- | --- | --- |
| **Asmx 輸出object** | | |
|  | **<string>xxx</string>** | **d:xxx**  **contentType: "application/json; charset=utf-8",** |
| **String** |  |  |
| **object** |  |  |
| **List<object>** |  |  |
| **總結** | 就是用object出來就會變成XML格式 | Object出來就會變成object; List<object>出來就是object<List> |

## Asmx 輸出object + 給參數

|  |  |  |  |
| --- | --- | --- | --- |
| **Asmx 輸出object + 給參數** | | | |
| 定義class request  定義傳進入的參數  [DataContract]  public class request  {  [DataMember]//如果沒有寫這個,就會出現//無效的 JSON 基本型別: x  public int x;  } | | | |
|  |  | **d:xxx**  **contentType: "application/json; charset=utf-8",** | **d:xxx**  **contentType: "application/json; charset=utf-8",**  **var postdata = { "x": 333 };**  **data: JSON.stringify(postdata),** |
| **object** |  | 無效的 JSON 基本型別: x |  |
| **List<object>** |  | 無效的 JSON 基本型別: x |  |
| **總結** | 就是用object出來就會變成XML格式 | Object出來就會變成object; List<object>出來就是object<List> |  |

常用語法

|  |  |
| --- | --- |
| Ajax呼叫 | $(document).ready(function () {  $.ajax({  url: "WebService1.asmx/HelloWorld2",  type: 'post',  success: function (data) {  console.log(data);  },  error: function (err) {  console.log("err:");  console.log(err);  }  });  }); |
| aspx | url: "WebForm4.aspx",  type: 'POST',  dataType: 'json',  data: { "x": 333 },//這樣可以  data: { x: 333 },//這樣可以  data: { x: '333' },//這樣可以 |
| Ajax傳變數 方法1 | data: { x: '333' },  data: { x: '1',b:'lalalalala' }, |
| Ajax傳變數 方法2 | data: 'x=' + '1'+'&b=' + 'lalalalala' , |
| Json解析object | success: function (data) {  console.log(data);  console.log(data.d);  console.log(data.d.a);  console.log(data.d.b);  }, |
| 拿<string>裡的東西 | var jqueryXml = $(data);  var x = jqueryXml.find('string').text(); |
|  | var myarray = $.parseJSON(x); |
| Json解析List<object> | $.each(myarray, function (i, item) {  var content = "我是第 " + i + " 個;叫做 " + item.a + "; 錯號叫做 " + item.b;  var li = "<li>" + content + "</li>";  $('#result').append(li);  }); |

把<string> 的皮拿掉

取d底面的值

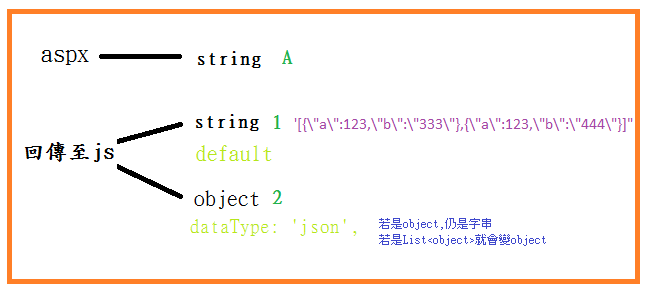
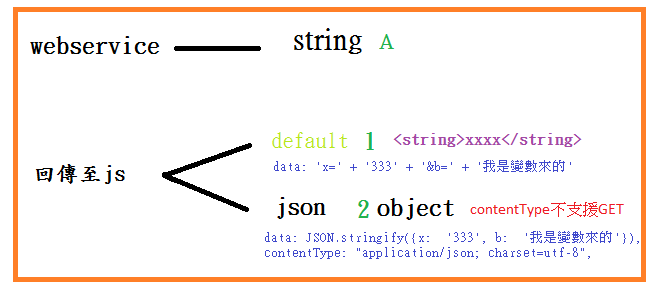
using System.Web.Script.Serialization;

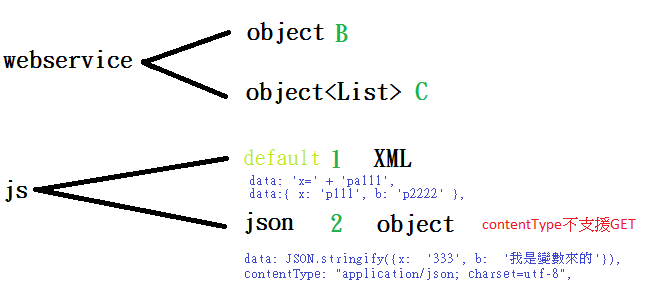
JavaScriptSerializer serializer = new JavaScriptSerializer();

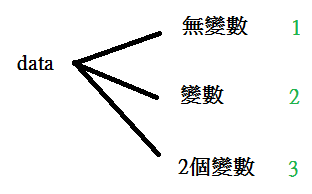
string ret = serializer.Serialize(lst);

<https://dotblogs.com.tw/hatelove/archive/2009/12/22/jqueryajax.aspx>

# xxxx







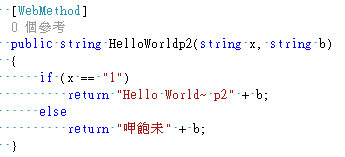
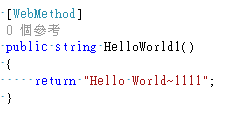
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Get1 | Get2 | Post1 | Post2 |
| XML | A | 無法辨認要求格式，因為 URL 未預期地以 /HelloWorld1 結束。  解決方法  <webServices>  <protocols>  <add name="HttpGet"/>  </protocols>  </webServices> | 無法辨認要求格式，因為 URL 未預期地以 /HelloWorldp1 結束。  解決方法  <webServices>  <protocols>  <add name="HttpGet"/>  </protocols>  </webServices> | O | O |
| B | O | O | O | O |
| C | O | O | O | O |
| json | A | 不允許這種方式 | 不允許這種方式 | 不允許這種方式 | 無效的 JSON 基本型別: x。 |
| B | 不允許這種方式 | 不允許這種方式 | 不允許這種方式 | 無效的 JSON 基本型別: x。 |
| C | 不允許這種方式 | 不允許這種方式 | 不允許這種方式 | 無效的 JSON 基本型別: x。 |
| 解決方法 |  |  | [DataContract] public class request  {  [DataMember]  Public int x;  } |  |  |

contentType不支援GET

|  |  |  |
| --- | --- | --- |
| A1 | 要轉成xml, 再取得string內的值, 再解析成json |  |
| A2 | 取出json字串 data.d, 再解析成json | contentType不支援GET |
| B1 | 轉出來的是XML格式 |  |
| B2 | Data.d.ID 這樣就讀出來了 |  |
| C1 | 轉出來的是XML格式 |  |
| C2 | Data.d[0].ID 這樣就讀出來了 |  |

|  |  |  |
| --- | --- | --- |
| A1 | 要轉成xml, 再取得string內的值, 再解析成json |  |
| A2 | 取出json字串 data.d, 再解析成json |  |
| B1 | 轉出來的是XML格式 |  |
| B2 | Data.d.ID 這樣就讀出來了 |  |
| C1 | 轉出來的是XML格式 |  |
| C2 | Data.d[0].ID 這樣就讀出來了 |  |

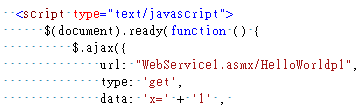
# A1



## Get1

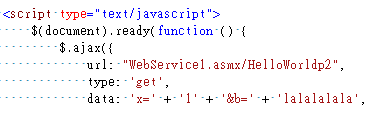


## Get2





## Get3



## Post1



## Post2

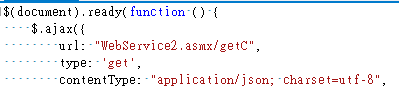


## Post3



# A2

## get : contentType不支援GET



<webServices>

<protocols>

<add name="HttpSoap"/>

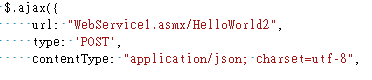
<add name="HttpPost"/>

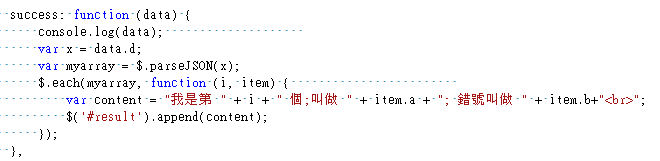
<add name="HttpGet"/>

</protocols>

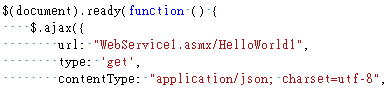
</webServices>

## post





## Get1-嘗試使用 GET 要求呼叫方法 'HelloWorld1'，不允許這種方式。



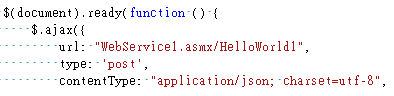
## Get2-嘗試使用 GET 要求呼叫方法 'HelloWorldp1'，不允許這種方式。



## Get3-嘗試使用 GET 要求呼叫方法 'HelloWorldp2'，不允許這種方式。



## Post1



## Post2-無效的 JSON 基本型別: x。

## Post3-無效的 JSON 基本型別: x。

# B1 輸出XML



## Get1

## Get2

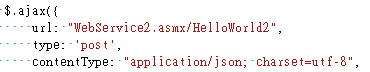
## Get3

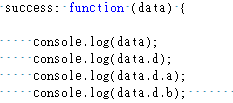
## Post1

## Post2

## Post3

# B2





## Get1-嘗試使用 GET 要求呼叫方法 'getB'，不允許這種方式

## Get2-嘗試使用 GET 要求呼叫方法 'getBp1'，不允許這種方式。

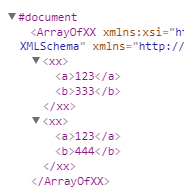
## Get3-嘗試使用 GET 要求呼叫方法 'getBp2'，不允許這種方式。

## Post1

## Post2-無效的 JSON 基本型別: x。

## Post3-無效的 JSON 基本型別: x。

# C1輸出XML



Get1

Get2

Get3

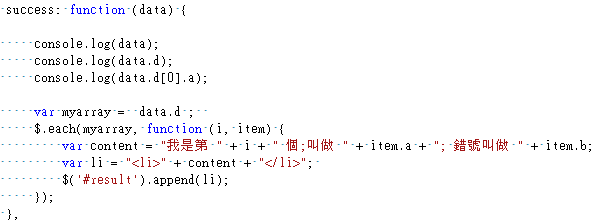
Post1

Post2

Post3

# C2





Get1

Get2

Get3

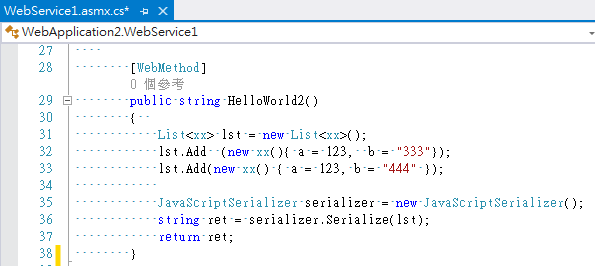
Post1

Post2

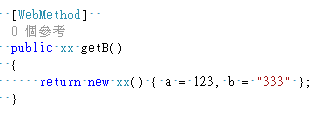
Post3

# Webservice

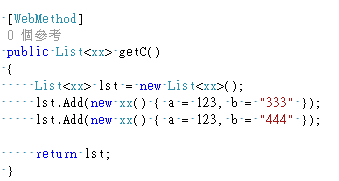
## ToString(List<object>)



## Object



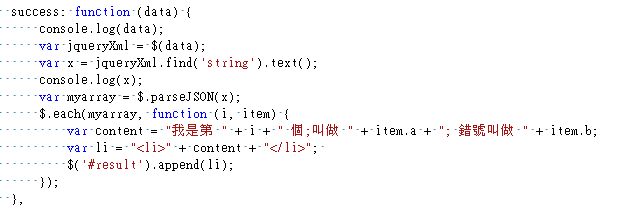
## List<object>



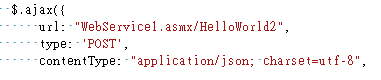
# Js

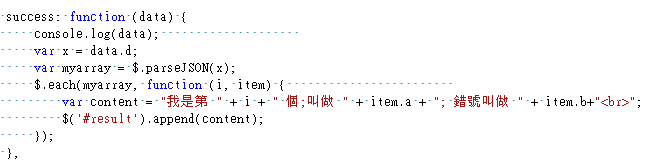
## XML





## Json





# webservice

<webServices>

<protocols>

<add name="HttpSoap"/>

<add name="HttpPost"/>

<add name="HttpGet"/>

</protocols>

</webServices>

# Ref - ajax參數說明

**參數說明 :**

ajax當中的參數：

**url(String)：**指定要進行呼叫的位址。

<https://dotblogs.com.tw/brooke/2016/08/26/164900>

## Async非同步

<script>

var result;

$.ajax({

url: "/Home/Index",

method: "get",

error: function () {

return result = "error";

},

success: function () {

return result = "success";

},

//async: false

});

alert(result);

</script>

如果以直線式思考第一直覺會覺得 result 的值一定是true或false，

但實際上執行的結果result會是undefined，這是為什麼呢？

其實是因為 $.ajax方法預設的async:true啟動非同步方法，

也就是說並不會等 $.ajax 執行完成才return，而是一開始就直接return了，

所以當然result的值當然會是undefined。

所以要解決這個問題，就是採用同步的方法來請求，所以我們可以將 async設成false。

## **data(Map)：傳參數**

用來將data送到server，如果data不是字串時則會自動轉成查詢字串，

如 {foo:["bar1", "bar2"]} 轉換為 '&foo=bar1&foo=bar2'。

可以藉由processData屬性設為false表示不會自動轉成查詢字串，

可以藉由traditional屬性設定序列化方式。



傳送至Server的資料，必須為Key/Value格式，GET請求中將附加在URL後面。

|  |  |
| --- | --- |
| 不同的寫法,但效果相同 | |
|  |  |
|  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | 所以若是沒用JSON.stringify,是object,若是有用,就是string. | |
|  | **Server function** |  | | **ajax.data** |
|  | aspx.string |  | | object |
|  | asmx.string |  | | object |
|  | asmx.object |  | | string |

## **type(String) 預設為get**

請求方式，POST/GET。(預設為GET)

## **dataType(String) dataType:json**預期Server傳回的資料類型

dataType is the form of data which i EXPECT to get from server

dataType: "json" only tells jQuery that you want it to parse the returned JSON

如果沒指定，jQuery會根據HTTP MIME Type自動選擇以responseXML或responseText傳入你的success callback。可選的資料類型有：

**xml**：傳回可用jQuery處理的XML。

**html**：傳回HTML，包含script tags。

**script**：傳回可執行的JavaScript。

**json**：傳回JSON。

**jsonp**：JSONP 格式。在URL加上?callback=?參數，並在Server端配合送回此jsonp callback。

**text**：傳回純文字字串。

## **success**請求成功時執行函式。

                function (data, textStatus) {

                    // data 可以是 xmlDoc, jsonObj, html, text, 但還是要參考datatype

                }

## **Success**

竟然嘛會駛安捏



## **error**請求失敗時執行函式。

function (xhr, ajaxOptions, thrownError) {

console.log(xhr.responseText);

//通常ajaxOptions或thrownError只有一個有值

}

* Xhr這裡主要有4個屬性：
  + readyState :當前狀態,0-未初始化，1-正在載入，2-已經載入，3-資料進行交互，4-完成。
  + status ：返回的HTTP狀態碼，比如常見的404,500等錯誤代碼。
  + statusText ：對應狀態碼的錯誤資訊，比如404錯誤資訊是not found,500是Internal Server Error。
  + responseText ：伺服器回應返回的文本資訊
* ajaxOptions：返回的是字串類型，表示返回的狀態，根據伺服器不同的錯誤可能返回下面這些資訊：
  + "timeout"（超時）,
  + "error"（錯誤）,
  + "abort"(中止),
  + "parsererror"（解析錯誤），
  + 還有可能返回空值。
* String errorThrown：也是字串類型，表示伺服器拋出返回的錯誤資訊，如果產生的是HTTP錯誤，那麼返回的資訊就是HTTP狀態碼對應的錯誤資訊，比如404的Not Found,500錯誤的Internal Server Error。

## **complete**請求完成時執行的函式(不論結果是success或error)。

                 function (XMLHttpRequest, textStatus) {

                   // the options for this ajax request

                 }

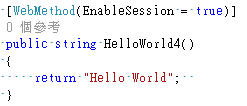
## contentType 傳來到client的值

contentType: "application/json; charset=utf-8"

contentType is the form of data which i send to the server

default: "application/x-www-form-urlencoded"。發送資訊至伺服器時內容編碼類型

這個好像是在定義傳進去的值

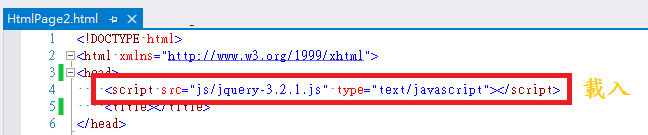


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | String | List<object> |  |
| Get  POST  都可以 |  | data: {x:3} | <string >字串</string> |  | var jqueryXml = $(data);  var x = jqueryXml.find('string').text(); |
| 只能用在POST,不能用在get | contentType: "application/json; charset=utf-8", | data: ‘{x:3}’  下了這行,傳進去的變數也要變成json格式進去 | d:字串 |  | var x =data.d; |

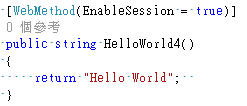
|  |  |
| --- | --- |
|  |  |
| $.ajax({  contentType: "application/json; charset=utf-8",  url: postUrl,  type: 'POST',  data: '{ act\_id: '+active\_id+' }',  async: false, |  |
| success: function (response) {  console.log(response);  console.log(response.d);  console.log(JSON.parse(response.d));  console.log(JSON.parse(response.d).calendar\_detail);  },  error: function(err){  console.log("err");  console.log(err);  } |  |

## contentType: "application/json; charset=utf-8"

### 建置html



### 2.建置webservice



### 呼叫

|  |  |
| --- | --- |
| XML | Json |
|  |  |
|  |  |
|  |  |
|  |  |

#### 2.1.呼叫回傳xml



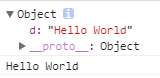


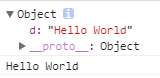
* 讀XML裡的值



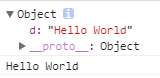
#### 2.2.呼叫回傳json

contentType: "application/json; charset=utf-8"





* 讀json裡的值



# Q:js先叫function1,function2,function3,結果,function3先來