**[Using ADO.NET Data Service – 客户端消费](http://www.cnblogs.com/zlgcool/archive/2008/11/30/1344577.html)**

ADO.NET Data service在前边已经谈了2 个部分，分别讲解了如何在server side构建一个ADO.NET Data Service和如何在Client Side消费Service，但是Data Service开发组仍然为我们提供了一个可以用AJAX Client library来消费Data Service的途径（您可以在CodePlex上下载到DataService Ajax Library，解压开后是一个名为DataServie.js的文件）。对于前边两部分不太熟悉的朋友可以参考：

         [Using ADO.NET Data Service – Server Side](http://www.cnblogs.com/zlgcool/archive/2008/10/15/1312231.html)

         [Using ADO.NET Data Service – Consuming](http://www.cnblogs.com/zlgcool/archive/2008/11/04/1326596.html)

在这篇里我们将会一起来熟悉一下如何来执行数据查询和相关操作。

**1.       RETRIEVE – ADO.NET Data Service AJAX Client Library**

ADO.NE Data Service AJAX Client Library是CodePlex上提供的对于客户端消费ADO.NET Data Service的JS类库。实际上执行的功能和上一章中我们讨论的在服务器端执行CRUD是相似的，只是换成了用AJAX的方式来执行而已，而且另一个区别是传递的数据都是JSON格式。

应用起来很简单，只需要引用DataService.js文件即可。

|  |
| --- |
| <div>      <asp:ScriptManager ID="ScriptManager1" runat="server">          <Scripts>              <asp:ScriptReference Path="~/Library/DataService.js" />          </Scripts>      </asp:ScriptManager>      <p>Filter the customers: <input type="text" id="txtFilter" style="width:120px" /><input type="button" value="Search" onclick="search();" /></p>      <div id="Result"></div>  </div> |

在pageLoad事件中我们希望查询出所有的Employees数据。

|  |
| --- |
| var northwindService = new Sys.Data.DataService("NorthwindService.svc");  northwindService.query("/Employees", cbSuccess, cbFailure); |

/Employees表示查询的URL，这里的所有操作都是通过URI来进行操作的。cbSuccess和cbFailure分别表示执行查询时成功或者失败分别对应的回调函数。通常cbSuccess函数是需要来做认真处理的，因为你得到的数据通过什么样的方式展现需要在这里控制。抛开任何模板不谈（当然你可以参考下一篇文章AJAX4.0中对模板的支持中找到新的答案。在另外一篇[jQuery和jTemplate构造客户端分页](http://www.cnblogs.com/zlgcool/archive/2008/11/28/1342959.html)中你也可以找到一些灵感==即便我们会在后边用到），我们想通过表格来展现数据：

|  |
| --- |
| function cbSuccess(result, context, operation) {              BuildTable(result);          }            function BuildTable(msg) {              var table = '<table><thead><tr class="title"><th>EmployeeID</th><th>Title</th>';              table += '<th>Name</th><th>Work Title</th><th>BirthDate</th><th>Country</th><th>City</th><th>Address</th></tr></thead><tbody>';              var i = 0;                for (var post in msg) {                  var row;                  if (i % 2 == 0) {                      row = '<tr>';                  }                  else {                      row = '<tr class="odd">';                  }                  row += '<td><a href="EmployeeView.aspx?EmployeeID=' + msg[post].EmployeeID + '">' + msg[post].EmployeeID + '</a></td>';                  row += '<td>' + msg[post].TitleOfCourtesy + " " + i + '</td>';                  row += '<td>' + msg[post].FirstName + ' ' + msg[post].LastName + '</td>';                  row += '<td>' + msg[post].Title + '</td>';                  row += '<td>' + msg[post].BirthDate.format("d") + '</td>';                  row += '<td>' + msg[post].Country + '</td>';                  row += '<td>' + msg[post].City + '</td>';                  row += '<td>' + msg[post].Address + '</td>';                  row += '</tr>';                    table += row;                  i += 1;              }                table += '</tbody></table>';              var result = document.getElementById("Result");              result.className = "";              result.innerHTML = table;          } |

唯一需要解释的便是返回的数据是以JSON集合的形式来表示的。遍历每一个对象并通过枚举其属性值来组建我们的table阵列--看起来很原始，但它真正达到了我们想要的结果，不是吗？

这个查询看起来很简单，但你已经看到了如何去获取数据并显示。再稍微扩展一下吧，做个搜索框，当点击的时候得到所有以你输入的字母开头的名字的员工列表。唯一的变化就是URI的不同：

|  |
| --- |
| function search() {      var filter = document.getElementById("txtFilter").value;      var result = document.getElementById("Result");      result.innerHTML = "";      result.className = "loading";        var northwindService = new Sys.Data.DataService("NorthwindService.svc");      northwindService.query("/Employees?$filter=startswith(FirstName,'" + filter + "') eq true", cbSuccess, cbFailure);  } |

2.       RETRIEVE – Using jQuery

jQuery是个很受欢迎的组件，它会让你体会到编写javascript的美妙，因为Microsoft已经将此置入了ASP.NET 4.0中。更为重要的是，jQuery可以和AJAX配合使用而没有任何冲突。当然另一个利用jQuery的好处之一便是在[jQuery和jTemplate构造客户端分页](http://www.cnblogs.com/zlgcool/archive/2008/11/28/1342959.html)一文中提到的jTemplate，在AJAX4.0 未发布之前，或许这很好的帮助了我们。

引用jQuery的库文件就不用说了，最主要的是如何去发出请求。记得在第一片文章那个里我们说过对于ADO.NET Data Service的请求关键字有GET, Update, Delete等，分清楚这个很重要。以下代码演示了如何去执行一个RETRIEVE查询：

|  |
| --- |
| $(document).ready(function() {   $('#Container').addClass("loading");   getValue();  });    function getValue() {   $.ajax({        type: "GET",        url: "http://localhost:5000/NorthwindService.svc/Employees?orderby=FirstName",        contentType: "application/json; charset=utf-8",        dataType: "json",        success: function(msg) {            // Render the resulting data, via template.            ApplyTemplate(msg);        },        error: function(xhr) {            showError(xhr);        }   });  } |

请求成功的回调函数仍然是执行了ApplyTemplate(msg)函数：我们将显示模板定义在一个HTML文件中，在生成时jTemplate会通过数据和定义的标记来RENDER. （有问题请参考[jQuery和jTemplate构造客户端分页](http://www.cnblogs.com/zlgcool/archive/2008/11/28/1342959.html)一文）。

3.       RETRIEVE – USING AJAX 4.0 (PREVIEW 3)

在AJAX 4.0 Preview 3中提供了对ADO.NET的支持和template的支持。这就像是替代了第一部分里便提供的AJAX Client Library一样，因为这里更为全面。AdoNetServiceProxy类是主要用来与ADO.NET Data Service通讯的类。MicrosoftAjaxAdoNet.js和**MicrosoftAjaxTemplates.js**需要引用到项目中—你可以从<http://www.codeplex.com/aspnet/Wiki/View.aspx?title=AJAX&referringTitle=Home>得到。

首先需要将AdoNetServiceProxy.js引入到项目中：

|  |
| --- |
| **<body   xmlns:sys="javascript:Sys"**  **xmlns:dv="javascript:Sys.UI.DataView"**  **sys:activate="\*">**  <asp:ScriptManager ID="ScriptManager1" runat="server">      <Scripts>          <asp:ScriptReference Path="Library/MicrosoftAjaxAdoNet.js" />          <asp:ScriptReference Path="Library/MicrosoftAjaxTemplates.js" />      </Scripts>      </asp:ScriptManager> |

接下来就可以执行query查询了。除了声明的类不同外（这里是AdoNetServiceProxy）其余的和第一部分中描述的方法类似。

|  |
| --- |
| function pageLoad() {              setupDataService();              queryService();          }          var dataService;          function setupDataService() {              dataService = new Sys.Data.AdoNetServiceProxy("NorthwindService.svc");              dataService.set\_timeout(60000);              dataService.set\_defaultFailedCallback(onFailure);              dataService.set\_defaultSucceededCallback(onSuccess);          }          function onSuccess(result, userContext, operation) {              var dataView = $find("employeeListView");              dataView.set\_data(result);                $get("employeeTemplate").style.display = "block";          }          function onFailure(result, userContext, operation) {              $get("errorStatus").value = result.get\_message() + ""r"tStatus Code: " + result.get\_statusCode() + ""r"tTimed Out: " + result.get\_timedOut();          }          function queryService() {              dataService.query("/Employees");          } |

但需要注意的是这里用到了客户端DataView。因为要使用Template功能来展现数据。如果你不想使用template，那么也可以将onSuccess中的方法调用设置为第一部分中的buildTable.

模板定义和jQuery的有些许不同，首先它定义在一个文件内，另外，扩展标记是使用{{}}而非$T。这里我们直接在定义模板时就驱动它自动执行查询来绑定。

|  |
| --- |
| <div id="employeeListView" sys:attach="dv"  dv:itemtemplate="employeeTemplate" dv:itemplaceholder="detailPh">      <div id="detailPh">         No selected data      </div>  </div>  <div>      <table>          <thead>              <tr class="title">                              <th>EmployeeID</th>                              <th>Title</th>                              <th>Name</th>                              <th>Work Title</th>                              <th>BirthDate</th>                              <th>Country</th>                              <th>City</th>                              <th>Address</th>              </tr>          </thead>          <tbody id="employeeTemplate" sys:attach="dv"              dv:datasource="{{ new Sys.Data.AdoNetDataSource() }}"              dv:serviceuri="NorthwindService.svc"              dv:query="Employees?$orderby=FirstName"              style="display:none">              <tr>                              <td>{{EmployeeID}}</td>                              <td>{{TitleOfCourtesy}}</td>                              <td>{{FirstName}}&nbsp;{{LastName}}</td>                              <td>{{Title}}</td>                              <td>{{BirthDate}}</td>                              <td>{{Country}}</td>                              <td>{{City}}</td>                              <td>{{Address}}</td>              </tr>          </tbody>      </table>  </div> |

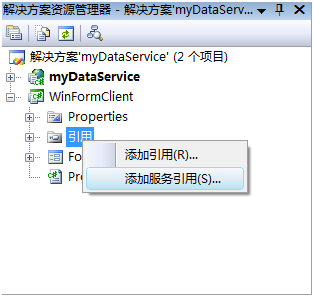
4.       操纵数据：

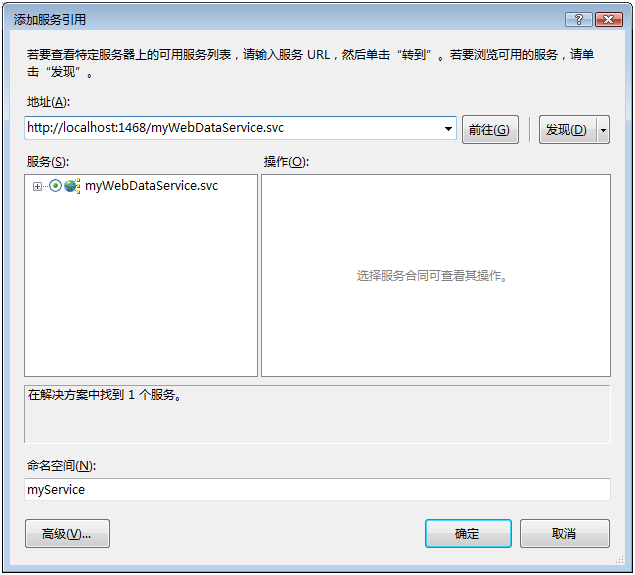
对于数据的操纵无外乎是针对AdoNetServiceProxy类的相关操作，除了数据是以JSON方式传递的外，看起来和在服务端消费数据服务没有太大的区别。直接上代码吧：

|  |
| --- |
| function pageLoad() {              setupDataService();              queryService();          }          var dataService;            function setupDataService() {              dataService = new Sys.Data.AdoNetServiceProxy("NorthwindService.svc");              dataService.set\_timeout(60000);              dataService.set\_defaultSucceededCallback(onSuccess);              dataService.set\_defaultFailedCallback(onFailure);          }            function onFailure(result, userContext, operation) {              alert(result.get\_message() + ""r"tStatus Code: " + result.get\_statusCode() + ""r"tTimed Out: " + result.get\_timedOut());          }            function queryService() {              var employeeId = <%= Request.QueryString["EmployeeID"] %>;              dataService.query("/Employees?$filter=EmployeeID eq " + employeeId);          }            function onSuccess(result, userContext, operation) {              if (result[0] != null) {                  $get("employeeID").value = result[0].EmployeeID;                  $get("txtFirstName").value = result[0].FirstName;                  $get("txtLastName").value = result[0].LastName;                  $get("txtTitle").value = result[0].TitleOfCourtesy;                  $get("txtBirthDate").value = result[0].BirthDate.format("d");                  $get("txtWorkTitle").value = result[0].Title;                  $get("txtCountry").value = result[0].Country;                  $get("txtCity").value = result[0].City;                  $get("txtAddress").value = result[0].Address;                  $get("txtNote").value = result[0].Note;              }          }            function onFailure(error, userContext, operation) {              alert(error);          }            function doUpdate() {              var employee = {                  EmployeeID:$get("employeeId").value,                  FirstName:$get("txtFirstName").value,                  LastName:$get("txtLastName").value,                  TitleOfCourtesy:$get("txtTitle").value,                  BirthDate:$get("txtBirthDate").value,                  Title:$get("txtWorkTitle").value,                  Country:$get("txtCountry").value,                  City:$get("txtCity").value,                  Address:$get("txtAddress").value,                  Note:$get("txtNote").value              };              dataService.update(employee, queryServiceUpdate);          }         function insertEmployee() {              var employee = {                  EmployeeID: 15,                  FirstName: $get("txtFirstName").value,                  LastName: $get("txtLastName").value,                  TitleOfCourtesy: $get("txtTitle").value,                  BirthDate: $get("txtBirthDate").value,                  Title: $get("txtWorkTitle").value,                  Country: $get("txtCountry").value,                  City: $get("txtCity").value,                  Address: $get("txtAddress").value              };              dataService.insert(employee, "Employees", onInserSuccess);          } |

# NET客启访问DataService

## 引用数据服务





## 查询

|  |
| --- |
| //查询  private void button1\_Click(object sender, EventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  myService.myDBEntities server = new WinFormClient.myService.myDBEntities(url);    foreach (var v in server.tabA)  {    Console.WriteLine("{0},{1},{2}", v.a, v.b, v.c);  }  //-  foreach (var v in server.tabX.Where(p=>p.x=="002"))  {    Console.WriteLine("{0},{1},{2}", v.x, v.y, v.z);  }    }    lzm ,2 ,5  wxd ,1 ,4  wxwinter ,3 ,6  002 ,8 ,llzzmm |

## 添加

|  |
| --- |
| //添加  private void button2\_Click(object sender, EventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  myService.myDBEntities server = new WinFormClient.myService.myDBEntities(url);    server.AddTotabA(new myService.tabA() { a = "wxdlzm", b = "333", c = "xxx" });    server.SaveChanges();  }  http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS15.png |

## 修改

|  |
| --- |
| //修改  private void button3\_Click(object sender, EventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  myService.myDBEntities server = new WinFormClient.myService.myDBEntities(url);  var v = server.tabA.Where(p => p.a == "wxdlzm").First();  v.b = "hello";  server.UpdateObject(v);  server.SaveChanges();  }    http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS16.png |

## 删除

|  |
| --- |
| //删除  private void button4\_Click(object sender, EventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  myService.myDBEntities server = new WinFormClient.myService.myDBEntities(url);  var v = server.tabA.Where(p => p.a == "wxdlzm").First();  server.DeleteObject(v);  server.SaveChanges();  }    http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS17.png |

## 异步查询

|  |
| --- |
| //异步查询    myService.myDBEntities server1;    //开始查询  private void button5\_Click(object sender, EventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  Uri urlE = new Uri("http://localhost:1468/myWebDataService.svc/tabX");  server1 = new WinFormClient.myService.myDBEntities(url);    server1.BeginExecutetabX>(urlE, new AsyncCallback(executed), null);  }  //返回结果  void executed(IAsyncResult obj)  {  var tp = server1.EndExecutetabX>(obj);    foreach (var v in tp)  {  Console.WriteLine("{0},{1},{2}", v.x, v.y, v.z);  }    } |

URL访问

DataService 支持URL方式访问

|  |  |
| --- | --- |
| 格式 | http://[Url]/[ServiceName]/[EntityName]/[NavigationOptions]?[QueryOptions] |
| Url: 数据服务所在的网址  ServiceName: 数据服务的名称  EntityName: Entity 名称或是 Service 方法名称  NavigationOptions: 查询 Entity 中关联的设定  QueryOptions: 查询的选项 |

## 运算符

|  |  |
| --- | --- |
| 运算符 | 说明 |
| and |  |
| or |  |
| not |  |
| eq | = |
| ne | != |
| lt | < |
| gt | > |
| le | <= |
| ge | >= |
| add | + |
| sub | - |
| mul | \* |
| div | / |
| mod | 余数 |
| () |  |

## 关键字

|  |  |
| --- | --- |
| expand | 类似于LINQ中的LoadOptions，以此来指定加载此对象相关的通过expand指定的对象。如果需要加载多个对象，用逗号分隔。 |
| orderby | 指定排序方式。语法为：$orderby=Field [ASC|DESC], [Field[ASC|DESC]] |
| Skip  Top | 类似于LINQ中的Skip(PageSize \* PageIndex).Take(PageSize)用来实现分页。 |
| Filter | 通过filter这个参数可以在URL里传递过滤条件。 |

## 函数

|  |  |
| --- | --- |
| bool substringof(string p0, string p1) |  |
| bool startswith(string p0, string p1) |  |
| int indexof(string arg) |  |
| string remove(string p0, int pos) |  |
| string remove(string p0, string find, string replace) |  |
| string substring(string p0, int pos, int length) |  |
| string toupper(string p0) |  |
| string concat(string p0, string p1) |  |
| bool endswith(string p0, string p1) |  |
| int length(string p0) |  |
| string insert(string p0,int pos, string p1) |  |
| string remove(string p0, int pos, int length) |  |
| string substring(string p0, int pos) |  |
| string tolower(string p0) |  |
| string trim(string p0) |  |
| int day(DateTime) |  |
| int month(DateTime) |  |
| int hour(DateTime) |  |
| int second(DateTime) |  |
| int minute(DateTime) |  |
| int year(DateTime) |  |
| double round(double) |  |
| decimal floor(decimal) |  |
| decimal round(decimal) |  |
| double ceiling(double) |  |
| double floor(double) |  |
| decimal ceiling(decimal) |  |

## 例1:直接访问Entity数据

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/tabA |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS25.png  由于返回的数据是ATOM格式的,如果浏览器支持该格式,会用阅读方式打开,如IE7用RSS阅读器方式打开  以下是返回的数据 |
| tabA  http://localhost:1468/myWebDataService.svc/tabA  2008-11-17T11:14:32Z  http://localhost:1468/myWebDataService.svc/tabA('lzm%20%20%20%20%20%20%20')  2008-11-17T11:14:32Z  lzm  2  5  http://localhost:1468/myWebDataService.svc/tabA('wxd%20%20%20%20%20%20%20')  2008-11-17T11:14:32Z  wxd  1  4  http://localhost:1468/myWebDataService.svc/tabA('wxdlzm%20%20%20%20')  2008-11-17T11:14:32Z  wxdlzm  333  xxx  http://localhost:1468/myWebDataService.svc/tabA('wxwinter%20%20')  2008-11-17T11:14:32Z  wxwinter  3  6 |

## 例2:使用条件

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/tabA?$filter=a eq 'wxd' |
| http://localhost:1468/myWebDataService.svc/tabA?$filter=(a eq 'wxwinter') and (b eq '3') |
| tabA  http://localhost:1468/myWebDataService.svc/tabA  2008-11-17T11:26:29Z  http://localhost:1468/myWebDataService.svc/tabA('wxd%20%20%20%20%20%20%20')  2008-11-17T11:26:29Z  wxd  1  4 |

## 例3:排序

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/tabA?$filter=a ne 'wxd' &$orderby=b |
| tabA  http://localhost:1468/myWebDataService.svc/tabA  2008-11-17T11:28:10Z  http://localhost:1468/myWebDataService.svc/tabA('lzm%20%20%20%20%20%20%20')  2008-11-17T11:28:10Z  lzm  2  5  http://localhost:1468/myWebDataService.svc/tabA('wxwinter%20%20')  2008-11-17T11:28:10Z  wxwinter  3  6  http://localhost:1468/myWebDataService.svc/tabA('wxdlzm%20%20%20%20')  2008-11-17T11:28:10Z  wxdlzm  333  xxx |

## 例4:分页

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/tabA?$filter=a ne 'wxd' &$orderby=b &$skip=2&top=2 |
| tabA  http://localhost:1468/myWebDataService.svc/tabA  2008-11-17T11:30:20Z  http://localhost:1468/myWebDataService.svc/tabA('wxdlzm%20%20%20%20')  2008-11-17T11:30:20Z  wxdlzm  333  xxx |

## 例5:函数的使用

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/tabA?$filter=a eq trim(' wxd ') |
| tabA  http://localhost:1468/myWebDataService.svc/tabA  2008-11-17T12:01:49Z  http://localhost:1468/myWebDataService.svc/tabA('wxd%20%20%20%20%20%20%20')  2008-11-17T12:01:49Z  wxd  1  4 |

# WEB方法

[WebGet] 使用 GET方式访问

[WebInvoke] 使用 POST/PUT/DELETE 方式访问

## 服务

|  |
| --- |
| public class myWebDataService : DataService<myDBEntities>  {    public static void InitializeService(IDataServiceConfiguration config)  {  config.SetEntitySetAccessRule("\*", EntitySetRights.All);  // \* :表示全部实体集  // EntitySetRights.All : 表示全部的操作权限    // config.SetServiceOperationAccessRule("getTabA", ServiceOperationRights.All);  config.SetServiceOperationAccessRule("\*", ServiceOperationRights.All);  // \* :表示全部实体集  // ServiceOperationRights.All : 表示全部的操作权限  }    [WebGet]  public IQueryable<tabA> getTabA(string a)  {  var v= CurrentDataSource.tabA.Where(p => p.a == a);  return v;  }    [WebInvoke]  public IQueryable<tabX> getTabX(string x)  {  var v = CurrentDataSource.tabX.Where(p => p.x == x);  return v;  }  } |

## 访问

方法名?参数='值

|  |
| --- |
| http://localhost:1468/myWebDataService.svc/getTabA?a='lzm' |
| getTabA  http://localhost:1468/myWebDataService.svc/getTabA  2008-11-17T02:23:29Z  http://localhost:1468/myWebDataService.svc/tabA('lzm%20%20%20%20%20%20%20')  2008-11-17T02:23:29Z  lzm  2  5 |

# 拦截器

查询拦截:当你仅仅想把具有某种状态或者特征的数据返回给客户端时，用拦截查询就可以实现

修改拦截:可以拦截提交到服务器的的数据更新操作：Add, Change ,Delete

## 查询拦截

|  |
| --- |
| public class myWebDataService : DataService<myDBEntities>  {    public static void InitializeService(IDataServiceConfiguration config)  {  config.SetEntitySetAccessRule("\*", EntitySetRights.All);  // \* :表示全部实体集  // EntitySetRights.All : 表示全部的操作权限  }    [QueryInterceptor("tabX")]  public System.Linq.Expressions.Expression<Func<tabX, bool>> query\_tabX()  {  return p => p.z != "wwxxdd"; //为真表示允许查询  }    } |
| 不使用QueryInterceptor的结果  http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS26.png    使用QueryInterceptor的结果  http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS27.png |

## 修改拦截器

|  |
| --- |
| public class myWebDataService : DataService<myDBEntities>  {    public static void InitializeService(IDataServiceConfiguration config)  {  config.SetEntitySetAccessRule("\*", EntitySetRights.All);  // \* :表示全部实体集  // EntitySetRights.All : 表示全部的操作权限  }  [ChangeInterceptor("tabX")]  public void change\_tabX(tabX en, UpdateOperations operation)  {  //-  if (operation == UpdateOperations.Add)  {  if (string.IsNullOrEmpty(en.y))  {  //如果[y]为空的处理代码  }  }  //-  if (operation == UpdateOperations.Change)  { }  //-  if (operation == UpdateOperations.Delete)  { }  }  } |

# Silverlight客启访问DataService

Silverlight 2.0 可以利用两种方法与DataServices交互：

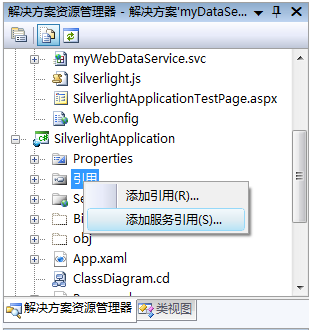
* HttpWebRequest 与 HttpWebResponse
* System.Data.Services.Client.DataServiceContext

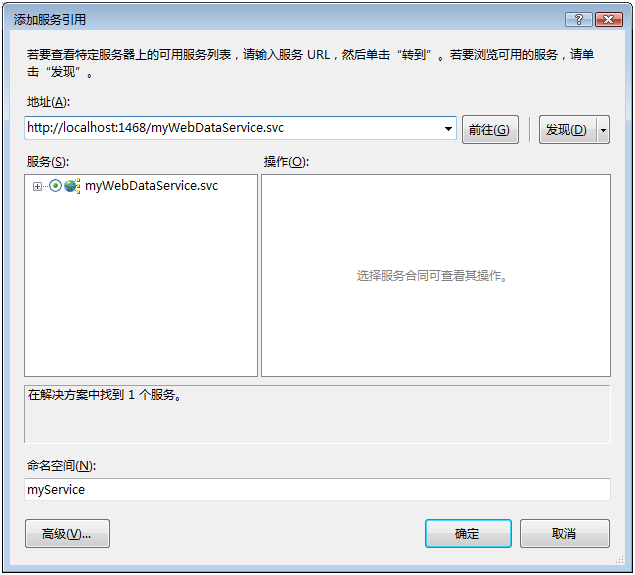
Silverlight 2.0 是使用异步方法来连接远程数据源。

LINQ 的 select ，要用 BeginExecute() 和 EndExecute()

LINQ 的 SaveChanges()，要用 BeginSaveChanges() 和 EndSaveChanges()

## 引用服务





## xaml页面

|  |
| --- |
| <StackPanel Loaded="StackPanel\_Loaded">  <data:DataGrid Name="dg">data:DataGrid>  <Button Content="DataServiceContext方式加载数据" Width="200" Click="loadData\_Click" />  <Button Content="WebClient方式加载数据" Width="200" Click="loadData2\_Click" />  <Button Content="添加" Width="200" Click="add\_Click" />  <Button Content="修改" Width="200" Click="edit\_Click" />  <Button Content="删除" Width="200" Click="del\_Click" />  StackPanel> |
| myService.myDBEntities server;  private void StackPanel\_Loaded(object sender, RoutedEventArgs e)  {  Uri url = new Uri("http://localhost:1468/myWebDataService.svc");  server = new myService.myDBEntities(url);  }  void saveChanges\_completed(IAsyncResult obj)  {    } |

## 查询(WebClient方式)

|  |
| --- |
| //WebClient方式加载数据  private void loadData2\_Click(object sender, RoutedEventArgs e)  {  Uri uri = new Uri("http://localhost:1468/myWebDataService.svc/tabA");  WebClient client = new WebClient();  client.OpenReadCompleted += new OpenReadCompletedEventHandler(client\_completed);  client.OpenReadAsync(uri);  }    void client\_completed(object sender, OpenReadCompletedEventArgs e)  {  if (e.Error == null)  {  XmlReader reader = XmlReader.Create(e.Result);  XDocument ls = XDocument.Load(reader);  XNamespace xmlns = "http://www.w3.org/2005/Atom";  XNamespace d = "http://schemas.microsoft.com/ado/2007/08/dataservices";  XNamespace m = "http://schemas.microsoft.com/ado/2007/08/dataservices/metadata";    var list = from x in ls.Descendants(xmlns + "entry")  select new myService.tabA  {  a= x.Descendants(d + "a").First().Value, b = x.Descendants(d + "b").First().Value, c = x.Descendants(d + "c").First().Value  };  dg.ItemsSource = list;  }  } |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS20.png |

## 查询

|  |
| --- |
| //加载  private void loadData\_Click(object sender, RoutedEventArgs e)  {  Uri urlE = new Uri("http://localhost:1468/myWebDataService.svc/tabX");  server.BeginExecutetabX>(urlE, new AsyncCallback(load\_completed), null);  }    void load\_completed(IAsyncResult obj)  {  var tp = server.EndExecutetabX>(obj);  this.dg.ItemsSource = tp.ToList();  } |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS21.png |

## 添加

|  |
| --- |
| //添加  private void add\_Click(object sender, RoutedEventArgs e)  {  server.AddTotabX(new myService.tabX() { x="007",z="sss",y="sss" });  server.BeginSaveChanges(new AsyncCallback(saveChanges\_completed), null);  } |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS22.png |

## 编辑

|  |
| --- |
| //编辑  private void edit\_Click(object sender, RoutedEventArgs e)  {  Uri urlE = new Uri("http://localhost:1468/myWebDataService.svc/tabX");  server.BeginExecutetabX>(urlE, new AsyncCallback(begin\_edit), null);  }  void begin\_edit(IAsyncResult obj)  {  var tp = server.EndExecutetabX>(obj).Where(p => p.x == "007 ").First();  tp.y = "hello";  server.UpdateObject(tp);  server.BeginSaveChanges(new AsyncCallback(saveChanges\_completed), null);  } |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS23.png |

## 删除

|  |
| --- |
| //删除  private void del\_Click(object sender, RoutedEventArgs e)  {  Uri urlE = new Uri("http://localhost:1468/myWebDataService.svc/tabX");  server.BeginExecutetabX>(urlE, new AsyncCallback(begin\_del), null);  }    void begin\_del(IAsyncResult obj)  {  var tp = server.EndExecutetabX>(obj).Where(p => p.x == "007 ").First();  server.DeleteObject(tp);  server.BeginSaveChanges(new AsyncCallback(saveChanges\_completed), null);    } |
| http://images.cnblogs.com/cnblogs_com/foundation/111808_0104_ADONETDataS24.png |