基于 Ajax 技术的 ASP.NET 数据分页①

屈武江

(大连海洋大学职业技术学院, 大连 116300)

摘 要: 在 Web 应用系统中经常在页面中浏览大量的数据, 在系统的开发过程中必然要遇到数据分页的问题, ASP.NET 提供了多种数据分页处理方式, 但在技术上都是采用同步传输方式, 这种方式在提取数据时用户等待的时间较长, 服务器负担较重. 本文在分析传统数据分页解决方案的基础上, 以学生信息浏览查询为实例, 详细介绍了在 ASP.NET 中基于 Ajax 技术实现数据分页的方式. 结论是基于 Ajax 技术的数据分页采用异步数据传输, 分页时通过局部更新数据, 无需重新加载整个页面, 大大减少了用户等待的时间, 减轻了服务器的压力负担, 提高了数据查询效率.

关键词: Ajax; 数据分页; 异步传输; Jquery; Json

Data Paging Based on the Ajax Technology in ASP.NET

QU Wu-Jiang

(Vocational & Technological College of Dalian Ocean University, Dalian 116300, China)

Abstract: A large amount of data is often browsed in the Web application system, so the problem of data paging is bound to occur in the process of system development. ASP.NET offers many data paging approaches, but technically all using the synchronous transmission mode. Using this way to extract data causes the user waiting longer and heavier burden on the server. Based on the analysis of traditional data paging solution, through an example of browsing and querying student information, this paper introduces in detail implementation of data paging based on the Ajax technology in ASP.NET. The conclusion is that data paging based on the Ajax technology using asynchronous data transmission, when paging through updating local data, without reloading the whole page, greatly reduces the users' waiting time, lightens the burden of the server,and improves data query efficiency.

Key words: Ajax; data paging; asynchronous transmission; Jquery; Json

在 WEB 应用系统中, 经常要查询大量的数据,因而分页浏览数据的功能是必不可少的. 纵观各种动态 WEB 应用系统, 数据分页的解决方案有多种, 如系统默认的 DataView 和 Dataset 组合分页查询、PageDataSource 对象用户自定义分页查询以及使用数据源实现无代码分页查询等, 但这些数据分页方式采用同步交互, 需要装载整个页面, 从而导致页面刷新, 网页加载速度缓慢.

随着技术的发展与进步, Ajax 逐渐成熟并广泛使用, 基于 Ajax 技术的分页查询, 只刷新页面所需数据, 其它页面元素不更新, 用户不用长时间等待就可以在页面中显示所需数据. 在ASP.NET中利用Ajax技术进行数据分页解决了传统数据分页网页显示较慢的问题, 广泛应用于各种 WEB 应用系统的数据查询功能[1].

1 ASP.NET常用数据分页解决方案

1.1 GridView 控件默认分页

在 ASP.NET 中 GridView 控件支持分页浏览, 设计时将GridView控件的AllowPaging属性设置为True, 服务器后台通过数据适配器从数据库提取所有符合条件的数据填充数据集 DataSet, 并将数据集绑定到 GridView 控件. 当用户单击分页导航按钮时, GridView

① 收稿时间:2013-02-27;收到修改稿时间:2013-04-01

控件自动根据当前页码从中选择指定页的数据, 显示在页面上. 这种方法编程简单, 适合数据量较少的情况, 如果数据较多时, 每次分页检索都要加载所有数据到 WEB 服务器, 影响 WEB 服务器的速度.

1.2 GridView 与 ObjectDataSource 组合实现数据分页

ObjectDataSource 控件对象相当于 ASP.NET 中提供的 SqlDataSource 控件. 该控件对象的 TypeName 属性指定要实例化来执行数据操作的类名.

ObjectDataSource 控件支持调用查询、更新、插入和删除等数据操作所关联的方法. ObjectDataSource 通过调用业务层类中对应 SelectMethod 方法来实现分页处理, 该方法要使用两个参数 MaximumRowsParameterName 和 StartRowIndexParameterName, 同时调用 SelectCount Method 方法, 得到查询数据总数, GridView 控件指定数据源属性为 ObjectDataSource 对象名并设置允许分页, ObjectDataSource 会自动根据两个参数 MaximumRows, StartRowIndex 来显示 GridView 当前页面的数据[2].

1.3 GridView 控件和 PageDataSource 实现自定义分页

GridView 具有 PageIndexChanged 事件, 在页面的后台定义返回 PageDataSource 对象的函数来实现分页数据提取以及分页设置, 并将其绑定到 GridView 控件中, 在页面分页导航按钮中调用数据绑定实现自定义分页.

1.4 ListView 和 DataPager 分页控件组合分页

DataPager 分页控件是.NET 3.5 新增控件, 目前只支持 ListView 控件, 如果想支持 GridView 可以通过自定义控件来实现. 应用 ListView 控件实现分页的基本原理是, 在后台代码中通过数据源对象提取数据并绑定 ListView 控件, DataPager 控件的 PagedConstrolID 属性设置为 ListView 控件的名称, PageSize 设置分页的行数, 分页控件所发生的事件调用业务层类提取数据并绑定.

1.5 Jquery+Ajax+Json 方式分页

Ajax 通过异步传输减少网络内容传输, Json 将输入的数据内容转换为序列化的 Json 格式, 然后利用 Jquery 内置的 Ajax 功能直接获得 Json 格式的数据,在客户端直接绑定或添加到数据控件中, 客户端程序将数据显示在页面上. 这种分页方式使客户端和服务器之间的数据通讯量很少, 不用刷新整个页面, 所以这种分页方式是一种理想的数据分页浏览方式.

2 相关技术介绍

2.1 Ajax 技术

Ajax 就是 Asynchronous JavaScript and XML, 中文含义是异步 JavaScript 和 XML, Ajax 并非缩写词, 而是由 Jesse James Gaiiett 创造的名词, 它是一种支持异步请求, 用来创建交互式动态网站的开发技术, 是一种全新的 WEB 应用程序设计模式. Ajax 并不是一项新技术, 而是多种技术的组合体. Ajax 主要包含的技术有 JavaScript、XHTML 和 CSS、DOM、XMLHttpRequest 和 XML 等.

在传统的 WEB 应用模式中, 数据传输是同步交互方式. 当用户在客户端浏览器向 Web 服务器发送 HTTP 请求信息时, WEB 服务器接收用户的请求, 并与数据库服务器进行数据交换, 再向发出请求的用户返回一个静态的 HTML 请求页面, 而在服务器处理后台查询数据时, 客户端浏览器处于等待状态, 如果数据量较大, 浏览器没有任何响应, 直至数据完全返回. 而 Ajax 的 WEB 应用模式, 数据传输采用异步交互方式, 当客户端浏览器发送请求时, 通过 JavaScript 方法调用 Ajax 引擎, Ajax 引擎再向 WEB 服务器发送 HTTP 请求, 通过调用 XMLHTTP 对象与服务器进行数据交换, 通过 DOM 解析处理 XML 文档并进行局部更新, 客户端与服务器之间的数据通信在后台运行, 这样, 用户发出 HTTP 请求后, 不必等待服务器的响应数据来刷新页面, 而是继续通过页面和服务器进行其他交互, Ajax 引擎则会自动选择适当的时机向服务器请求数据并将返回的数据显示在客户端. 这种应用模式, 不仅减轻了服务器的负担, 而且减少了用户等待的时间. 图 1 所示为传统 WEB 应用模式, 图 2 所示为基于 Ajax 的 WEB 应用模式[3].

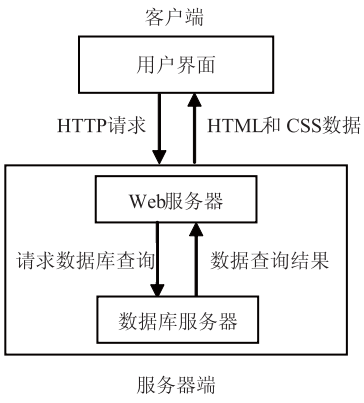

图 1 传统 Web 应用模式

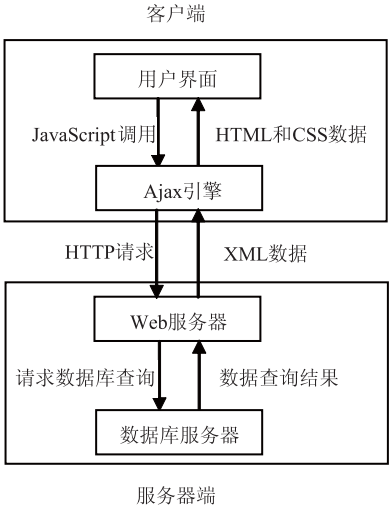

图 2　基于 Ajax 的 Web 应用模式

2.2 JSON 数据

JSON 是 JavaScript Object Notation 的缩写, 它是一种轻量级的数据交换格式, 易于阅读和编写, 同时也易于机器解析和生成. 它基于 ECMA262 语言规范(1999-12 第三版)中 JavaScript 编程语言的一个子集. JSON 采用与编程语言无关的文本格式, 但是也使用了类 C 语言(包括 C, C++, C#, Java, JavaScript, Perl, Python 等)的习惯, 这些特性使 JSON 成为理想的数据交换格式[4].

2.3 Jquery 简介

Jquery 是一个兼容多浏览器的 JavaScript 库, 核心理念是 Write less, do more(写的更少,做的更多). Jquery 是免费、开源的, 使用 MIT 许可协议. 应用 jQuery 库可以轻松获取文档中的元素, 能够改变文档的内容, 响应用户的交互操作, 为页面添加动态效果, 而无需刷新页面, 就可以从服务器获取信息, 简化常见的 JavaScript 任务, 同时 Jquery 提供 API 让开发者编写插件, 其模块化的使用方式使开发者可以很轻松的开发出功能强大的静态或动态网页[5].

3　基于Ajax技术实现数据分页的设计

通过 ASP.NET 几种常用数据分页解决方案和 Ajax 相关技术的介绍, 下面以学生信息浏览功能为实例, 介绍应用 Ajax 实现 WEB 浏览查询系统数据分页的设计.

本实例的设计环境是: 软件开发工具为 Visual Studio 2008, 开发语言为 C#, 数据库管理系统为 ACCESS, Jquery 版本为 1.4.4.

学生信息分页浏览页面运行结果如图 3 所示.

图 3　学生信息浏览页面运行结果

3.1 学生信息浏览数据分页的设计思路

客户端用户在浏览器打开浏览学生信息页面时, 首先初始化页面, 显示学生信息的第 1 页数据, 当用户单击“下一页”或导航页码按钮提交分页请求指令时, 由客户端 JavaScript 脚本截获并异步发送到 Web 服务器, Web 服务器接收数据分页请求, 调用业务层数据操作通用类获取用户请求的分页数据并转换成 JSON 格式, JSON 格式数据被异步发回客户端 JavaScript 脚本, 调用函数在浏览器后台动态局部刷新用户界面, 实现分页操作. Ajax 实现数据分页的基本流程如图 4 所示[6].

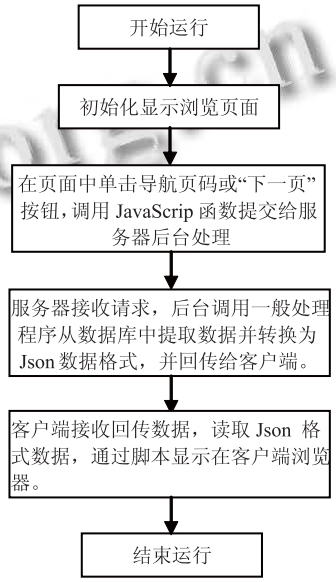

图 4　Ajax 实现数据分页基本流程

3.2 学生信息浏览数据分页功能的设计

3.2.1 数据操作业务层类的设计

为了增加程序的通用性和可移植性, 建立实现数

据库连接和数据查询操作的业务层类 DBOpertion, 并且将数据库连接串代码存放在 Web.Config 文件, 该文件是一个 XML 文件, 可以随时更改文件的内容, 修改后立即生效, 不必重新启动 Web 服务器.

在 DBOpertion 类中, 先声明全局连接对象, 构造函数实现从 Web.Config 中取得连接串并设置连接对象的连接串属性. 在该类中定义 CreateDataTable 方法, 返回 DataTable 对象, 功能是使用数据适配器从数据库中查询传递给 strsql 参数的 SQL 查询语句, 调用数据适配器的 Fill 方法填充数据集, 在该类中声明 DataTable 对象的表结构, 通过循环将数据集查询结果添加到 DataTable 对象返回, 并设置 RecordCount 取得记录总数, 关键代码如下:

```
……
public    DataTable CreateDataTable(string strsql)
{
    oledbconn.Open();//打开连接
    DataSet ds = new DataSet();
    OleDbDataAdapter adp = new OleDbDataAdapter(strsql, oledbconn);
    adp.Fill(ds);
    //定义 DataTbale 对象各列的字段名和数据类型
    DataTable dtable = new DataTable();
    dtable.Columns.Add(new DataColumn("Xh", typeof(string)) { DefaultValue = "" });
    ……//定义数据表各列的字段信息
    for (int i = 0; i <= ds.Tables[0].Rows.Count - 1; i++)
    {
        ……//循环体将查询的数据添加到数据表对象
    }
HttpContext.Current.Cache.Insert("RecordCount", dtable.Rows.Count);
    RecordCount = dtable.Rows.Count;
    return dtable;
    }}
```

3.2.2 客户端 JavaScipt 脚本读取返回数据呈现用户界面

在该 ASPX 文件客户端 HTML 代码中, 从外部链接两个 JavaScrip 脚本文件, 即 Jquery, 两个样式文件, 分别设置数据显示表格和分页导航的样式. 代码中首先初始分页的参数, 即页面索引初始值、每页显示记录条数、分页导航按钮文本、分页导航的数字按钮个数、两侧分页按钮的条目数等. 然后定义 PageCallback()函数调用 Init()函数提交到一般处理程序请求数据, 下方的 HTML 代码用来显示数据表头和分页数据. 关键 JavaScrip 客户端脚本如下:

```
<script type="text/javascript">
var pageIndex = 0; //定义页面索引初始值
var pageSize =10; //每页显示记录条数
$(function () {
    Init(0); //初始化页面索引为 0 的第 1 页表格数据
    $("#Pagination").pagination(<%=pageCount %>, {
        callback: PageCallback,
        ……//初始化分页参数
    });
    function PageCallback(index, jq) //定义调用分页函数
     {Init(index);}
function Init(pageIndex) {
        $.ajax({
            type: "POST",
            dataType: "json",
            url: 'Json.aspx',
            data: "type=display&random=" + Math.random() + "&pageIndex=" + (pageIndex + 1) + "&pageSize=" + pageSize,
            error: function () { alert('error data'); }, //异常处理, 发生错误时提示错误信息
            success: function (data) {
                $("#Result tr:gt(0)").remove();
                var json = data;
                 var htmldata =
$.each(json.data, function (index, jsonitem) {
                    var xh = jsonitem.Xh;
                    ……
                    htmldata += "<tr><td>" + xh + "</td><td>" + xm + "</td><td>" + xb + "</td><td>"+csrq+"</td><td>"+dh+"</td><td>"+yb+"</td><td>"+jtdz+"</td></tr>";
```

```
                });
                $("#Result").append(htmldata);
    </script >
    ……
```

3.2.3 读取 Json 格式数据的客户端后台代码

客户端后台代码负责页面加载时通过调用 GetJsonString()和 GetJsonString()方法读取 Json 格式数据,而在方法中间接调用一般处理程序将用户请求的数据自动转换为 Json 格式,返回给调用方法. 关键代码如下:

```
    ……
    //后台获取返回数据的方法
    public static string GetJsonString(string relativePath, string data)
    {   string RequestUrl = GetRequestUrl(relativePath, data);
        try{
            WebRequest RequestObject = WebRequest.Create(RequestUrl);
            RequestObject.Method = "GET";
            StreamReader JsonStreamData = new StreamReader(RequestObject.GetResponse().GetResponseStream());
            string JsonObjectSt = JsonStreamData.ReadToEnd();
            return JsonObjectSt;}
        catch{
            return string.Empty;}
    }
    public static string GetRequestUrl(string relativePath, string data)
    {
        string absolutePath = HttpContext.Current.Request.Url.AbsoluteUri;
        string hostNameAndPort = HttpContext.Current.Request.Url.Authority;
        string applicationDir = HttpContext.Current.Request.ApplicationPath;
        StringBuilder sbRequestUrl = new StringBuilder();
sbRequestUrl.Append(absolutePath.Substring(0, absolutePath.IndexOf(hostNameAndPort)));
        sbRequestUrl.Append(hostNameAndPort);
        sbRequestUrl.Append(applicationDir);
        sbRequestUrl.Append(relativePath);
        if (!string.IsNullOrEmpty(data))
        {
            sbRequestUrl.Append("?");
            sbRequestUrl.Append(data);
        }
        return sbRequestUrl.ToString();
    }
```

3.2.4 转换数据格式实现分页处理的一般程序

后台处理一般程序的主要功能是实现从数据库中读取数据并转换为 Json 数据格式, 方法 GetPagedTable 进行数据分页配置.

```
//从数据库中读取数据并转换为 Json 数据格式
        ……
        DBOperation getdata = new DBOperation();
        string strsql="select f_xh as Xh,f_xm as Xm,f_xb as Xb,f_csrq as Csrq,f_dh as Dh,f_yb as Yb,f_jtdz as Jtdz from [t_student]";
        DataTable dt = getdata.CreateDataTable(strsql);
        DataTable PagedDataTable = GetPagedTable(dt, pageIndex, PageSize);
        List<Struct_S> list = new List<Struct_S>();
        foreach (DataRow dr in PagedDataTable.Rows)
            {
                Struct_S student = new Struct_S();
                student.Xh = dr["Xh"].ToString();
                ……
                list.Add(student);
            }
        string jsondata = new JavaScriptSerializer().Serialize(list);
        StringBuilder Builderstring = new StringBuilder();
        Builderstring.Append("{");
Builderstring.Append("\"recordcount\":" + RecordCount + ",");
        Builderstring.Append("\"data\":");
        Builderstring.Append(jsondata);
```

```
            Builderstring.Append("}");
            Response.ContentType = "application/json";
 Response.Write(Builderstring.ToString());
            ……
 //定义分页配置的方法
    public static DataTable GetPagedTable(DataTable dtable, int PageIndex, int PageSize)
    {
        if (PageIndex == 0)
            return dtable;
        DataTable NewDataTable = dtable.Copy();
        NewDataTable.Clear();
        int rowbegin = (PageIndex - 1) * PageSize;
        int rowend = PageIndex * PageSize;
        if (rowbegin >= dtable.Rows.Count)
            return NewDataTable;
        if (rowend > dtable.Rows.Count)
            rowend = dtable.Rows.Count;
            ……//部分代码略
    }
```

4 结论

传统的 WEB 应用系统数据分页采用同步数据传输, 这种 Web 应用模式存在数据查询效率低的缺点. 在介绍传统数据分页方式和 Ajax 相关技术的基础上, 以学生信息浏览为实例详细论述了 ASP.NET 采用 Ajax、Jquery 与 Json 相结合的异步传输数据快速分页方式, 这种数据分页方式大大降低了网络通信数据量, 节省了网络带宽, 提高用户访问的响应速度, 服务器的压力得以降低. 因此基于 Ajax 技术的数据分页方式适合数据量较大的 WEB 应用系统的数据查询功能.

参考文献

1 刘红坤.基于 Ajax 和 PHP 数据分页的实现.计算机系统应用,2012,21(2):218–220.

2 张欣,朱占立,李永.NET 框架下基于 AJAX 的 Web 数据分页研究.电脑知识与技术,2008,4(1):30–32.

3 周杨.AJAX 应用的典型设计模式.计算机系统应用,2011,20(1):128–132.

4 屈展,李婵.JSON 在 Ajax 数据交换中的应用研究.西安石油大学学报,2011,26(1):85–98.

5 卞艺杰,邹银马,赵喆.基于 S2Si+ jQuery 的高校短信平台设计与实现.中国制造业信息化,2012,41(1):71–74.

6 李卿,楼新远.基于 AJAX 的数据分页的设计与实现.成都信息工程学院学报,2008,23(2):191–194.

(上接第 184 页)

background subtraction. Proc. of the 17th International Conference of Pattern Recognition. Cambrige, UK: IEEE, 2004:28–31.

7 Zivkovic Z, van der Heijden F. Effcient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognition Letters,2006,27(7):773–780.

8 Yang H, Tan Y, Tian J, Liu J. Accurate dynamic scene model for moving object detection. Int Conf on Image Processing (ICIP2007),2007,6:157–160.

9 杨涛,李静,潘泉,程咏梅.一种基于多层背景模型的前景检测算法.中国图象图形学报,2008,13(7):1303–1308.

10 常晓夫,张文生,董维山.基于多种类视觉特征的混合高斯背景模型.中国图象图形学报,2011,16(5):829–834.

11 Flavell JH. Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. American Psychologist,1979,34(10):906–911.

12 Efklides A, Misailidi P. Trends and Prospects in Metacognition Research. New York: Springer. 2010.