第 13 章　ASP.NET AJAX 无刷新数据处理技术

AJAX（Asynchronous JavaScript and XML，异步 JavaScript 和 XML）是一种创建互动式网页应用的网页开发技术。在传统的应用程序中，用户在客户端填写表单，然后提交表单时向服务器发送一个请求。服务器端接到表单后，处理并返回一个新的表单。这种交互方式比较浪费带宽资源，因为也许前后表单内容变化并不大。如果表单数据太大，客户端就会等待比较长的时间。在这种情况下，就引入了 AJAX 技术。

AJAX 可以只向服务器发送并取回所必须修改的数据，它在客户端采用 JavaScript，处理来自服务器的响应。如果服务器和浏览器之间交换数据越少，客户端就能接受到更快的响应。同时，很多的处理工作都在客户端机器上完成，所以 Web 服务器的处理时间也减少了。

AJAX 应用程序的优势在于以下几个方面。

- 通过异步模式，大大提升了用户体验。
- 优化了浏览器和服务器之间的传输，减少了大量不必要的数据往返，减少了带宽占用。
- AJAX 引擎在客户端运行，承担了部分本来由服务器需要承担的工作，从而减少了超大用户量下的服务器负担。

【本节示例参考：\源代码\C13\AjaxSample】

13.1　ASP.NET AJAX 概述

使用 ASP.NET 中的 AJAX 功能，可方便快捷地创建响应能力快的网页，并提供丰富的用户体验效果。AJAX 的实现其实来自其客户端脚本库。这些库将跨浏览器的 JavaScript 和 DHTML 技术结合在一起，并与基于 ASP.NET 服务器的开发平台集成。通过使用 AJAX 功能，可以改进用户体验并提高 Web 应用程序的效率。

13.1.1　ASP.NET 中 AJAX 功能

使用 ASP.NET 中的 AJAX 功能，可以生成丰富的 Web 应用程序。与完全基于服务器的 Web 应用程序相比，这些应用程序具有很多优点。

- 增强效率。这是因为网页的大部分处理工作是在浏览器中执行的。
- 熟悉 UI 元素，如进度指示器、工具提示和弹出窗口等。
- 部分页更新，只刷新已发生更改的网页部分。

- 客户端与用于 Forms 身份验证的 ASP.NET 应用程序服务、角色和用户配置文件的集成。
- 自动生成的代理类，可简化从客户端脚本调用 Web 服务方法的过程。
- 对大部分流行和常用的浏览器的支持，包括 Microsoft Internet Explorer、Mozilla Firefox 和 Apple Safari 等。

13.1.2 ASP.NET 中的 AJAX 功能的结构

ASP.NET 中的 AJAX 由两部分组成：客户端脚本库和服务器组件。这两个组成部分集成在一起，为开发人员提供了可靠的开发框架。如图 13-1 显示了客户端脚本库和服务器组件中的所有功能。

图 13-1　ASP.NET AJAX 客户端和服务器结构

注意：除 ASP.NET 中的 AJAX 功能之外，还可以使用 ASP.NET AJAX 控件工具箱和 Microsoft ASP.NET CTP 版本中的功能。

图 13-1 中显示了基于客户端的 Microsoft AJAX Library 功能，主要包含对创建客户端组件、浏览器兼容性以及网络和核心服务的支持。还显示了基于服务器的 AJAX 功能，主要包含脚本支持、Web 服务、应用程序服务和服务器控件。

13.1.3 AJAX 客户端结构

客户端结构包括用于组件支持、浏览器兼容性、网络和核心服务的库。

1．组件

客户端组件在服务器中可以进行一定的处理操作，而不需要回发。这些组件分为以下 3 类。

- 组件：封装代码的非可视化对象（如计时器 Timer）。
- 行为：可扩展现有 DOM 元素的基本行为。

- 控件：具有自定义行为的新 DOM 元素。

要使用什么组件类型，取决于客户端行为的类型。例如，可使用附加到当前文本框的行为创建该文本框的水印效果（AJAX 中提供了这样的组件）。

2．浏览器兼容性

浏览器兼容性，就是针对最常用的浏览器（包括 Microsoft Internet Explorer、Mozilla Firefox 和 Apple Safari）提供 AJAX 脚本兼容。这使开发人员能够针对各种受支持的服务器，编写相同的脚本。

3．网络

网络层一般处理浏览器中的脚本与应用程序间的通信，还负责管理异步远程方法调用。网络层还提供对在客户端脚本中，访问基于服务器的 Forms 身份验证、角色信息和配置信息的支持。只要应用程序有访问 Microsoft AJAX Library 的权利，在不使用 ASP.NET 创建的 Web 程序中也可获得此支持。

4．核心服务

ASP.NET 中的 AJAX 客户端脚本库由 JavaScript（.js）文件组成。这些文件提供用于面向对象开发的功能。感兴趣的读者可以研究一下这些脚本库，会重新认识 JS 的用途。下面的核心服务是客户端结构的一部分。

- 针对 JavaScript 的面向对象扩展。
- 一个基类库，包含字符串生成器和扩展的错误处理等组件。
- 对 JavaScript 库的支持，这些库会嵌入到程序集中。在程序集中嵌入 JavaScript 库，将使部署应用程序更容易。

13.1.4　AJAX 服务器结构

AJAX 服务器由负责界面 UI 的 ASP.NET Web 服务器控件和组件组成。这些服务器还管理序列化、验证、控件扩展性等。服务器中还包含一些 ASP.NET Web 服务，这就允许客户端访问用于 Forms 身份验证、角色和用户配置文件的 ASP.NET 应用程序。

1．脚本支持

服务器端可以给客户端发送一些 JS 脚本，实现真正的 AJAX 功能。除了 AJAX 提供的一些 JS 外，还可以自定义客户端脚本，这样就能用 AJAX 来管理这些自定义的脚本，同时还可以把这些脚本嵌入到程序集中使用。

ASP.NET AJAX 还包含用于发布模式和调试模式的模型。发布模式提供错误检查和异常处理，并优化性能；调试模式提供可靠的调试功能，如类型和参数检查。

2．本地化

ASP.NET AJAX 对嵌入到程序集中的.js 文件提供本地化支持，所以 AJAX 可自动针对特定语言和区域，提供本地化的客户端脚本和资源。

3．Web 服务

AJAX 允许用客户端脚本调用 ASP.NET Web 服务（.asmx）。请求的脚本引用将自动添加到当前页，随后自动生成 Web 服务代理类。

也可以不使用 AJAX 服务器控件，直接访问 ASP.NET Web 服务。这需要手动在页中添加对 Microsoft AJAX Library、脚本文件和 Web 服务本身的引用。在运行时，ASP.NET 自动生成可用于调用这些服务的代理类。

4．服务器控件

AJAX 服务器控件由服务器和客户端两类代码组成，组合后可生成多功能的客户端行为。在 Visual Studio 2008 中，常见的 ASP.NET AJAX 服务器控件如下。

- ScriptManager：管理客户端组件、部分页呈现、本地化、全球化和自定义用户脚本的脚本资源。若要使用 UpdatePanel、UpdateProgress 和 Timer 控件，则需要 ScriptManager 控件。
- UpdatePanel：刷新部分页内容，而不是通过回发来刷新整个页面。
- UpdateProgress：提供在更新过程中的提示。
- Timer：按定义的时间间隔重复执行某个操作。

13.2 创建 AJAX 应用程序

本节创建使用 ASP.NET 的 AJAX 功能的基本应用程序。读者可以了解到有关 ASP.NET 的 AJAX 功能的更多信息，将知道这些功能旨在解决哪些技术问题，以及以下介绍性文档将涉及哪些重要的 AJAX 组件。

13.2.1 创建 AJAX 的网页

创建一个 ASP.NET 网站后，在页面中加入日期控件和一个下拉列表框。根据下拉列表框选择的不同，日期控件背景变为不同的颜色。操作步骤如下。

（1）打开“新建网站”窗口，在“项目类型”列表框中，选中“Visual C#项目”。在“模板”列表框中选中“ASP.NET 网站”，在“位置”文本框中，将项目的名称设为 AjaxSample。单击“确定”按钮，则新建了一个名称为 AjaxSample 的项目。

（2）在项目中，右击“解决方案资源管理器”面板，在弹出的快捷菜单中，单击“添加”→“Web 窗体”命令，则弹出“添加新项”窗口。在“名称”文本框中，将名称更改为“FirstAJAXApp.aspx”。

（3）在窗体上依次添加 UpdatePanel 控件、Calendar 控件、DropDownList 控件、ScriptManager 控件，UpdatePanel 控件依赖于 ScriptManager 控件来管理部分页更新。其设计视图如图 13-2 所示。这些 Ajax 控件都在工具箱的 AJAX Extensions 组中。

图 13-2　设计视图

设计界面的源代码如代码 13-01 所示。

代码 13-01　日期控件背景变色

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>无标题页</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>                                    <!--下面是局部刷新内容-->
        <asp:UpdatePanel ID="UpdatePanel1" runat="server">
            <ContentTemplate>
                <asp:ScriptManager ID="ScriptManager1" runat="server">
                </asp:ScriptManager>
                <asp:Calendar ID="Calendar1" runat="server"></asp:Calendar>
                <br />
                <asp:DropDownList ID="DropDownList1" runat="server" AutoPostBack="True"
                    onselectedindexchanged="DropDownList1_SelectedIndexChanged"
style="width: 57px">
                    <asp:ListItem Value="Red">红色</asp:ListItem>
                    <asp:ListItem Value="Blue">蓝色</asp:ListItem>
                    <asp:ListItem Value="Yellow">黄色</asp:ListItem>
                    <asp:ListItem Value="Pink">粉色</asp:ListItem>
                    <asp:ListItem Value="Green">绿色</asp:ListItem>
                    <asp:ListItem Value="Orange">橘色</asp:ListItem>
                    <asp:ListItem Selected="True" Value="White">白色</asp:ListItem>
                </asp:DropDownList>
            </ContentTemplate>
        </asp:UpdatePanel>              <!--以上是局部刷新内容-->
    </div>
    </form>
</body>
</html>
```

后台的功能代码如代码 13-02 所示，本例只是将日历的背景色作了修订。

代码 13-02　日期控件背景变色

```
public partial class FirstAJAXApp : System.Web.UI.Page
{
    protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
    {
        Calendar1.DayStyle.BackColor =
System.Drawing.Color.FromName(DropDownList1.SelectedItem.Value);
    }
}
```

（4）在浏览器中查看 FirstAJAXApp.aspx，效果如图 13-3 所示。在下拉列表框中选择“橘色”后，效果如图 13-4 所示。

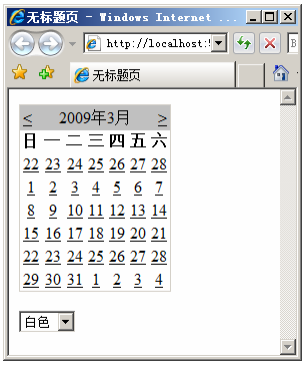

图 13-3　浏览 FirstAJAXApp.aspx 效果

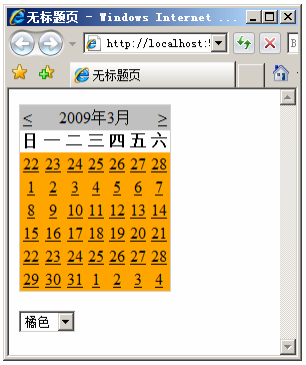

图 13-4　选择“橘色”后效果

13.2.2　使用 Timer 控件

Timer 控件按定义的时间间隔执行回发。如果将 Timer 控件用于 UpdatePanel 控件，则可以按定义的时间间隔启用部分页更新，还可以使用 Timer 控件来发送整个页面。

1．Timer 控件方案

当要执行以下操作时，可以使用 Timer 控件。

- ❑ 定期更新 UpdatePanel 控件的内容，此时无需刷新整个网页。
- ❑ Timer 控件回发时运行服务器上的代码。
- ❑ 按定义的时间间隔将整个网页发布到 Web 服务器上。

2．背景

Timer 控件属于服务器控件，其允许将一个 JavaScript 组件嵌入到网页中。当经过指定的时间间隔后，该组件将从浏览器启动回发。Timer 控件通过 Tick 事件，进行一些逻辑处理。

注意：使用 Timer 控件时，必须在网页中包括 ScriptManager 类的示例。

如果不同的 UpdatePanel 控件以不同的时间间隔更新，则可以在网页上包含多个 Timer 控件。可以在 UpdatePanel 控件内部使用 Timer 控件，也可以在 UpdatePanel 控件外部使用 Timer 控件。

注意：将 Timer 控件的 Interval 属性设置为一个较小值，会产生发送到 Web 服务器的大量通信。使用 Timer 控件可以仅按所需的频率刷新内容。

13.2.3　使用 Timer 控件创建应用程序

在本节中，将使用 3 个 ASP.NET AJAX 服务器控件（ScriptManager 控件、UpdatePanel 控件和 Timer 控件）按固定的时间间隔更新部分网页。通过将这些控件添加到网页上，可消除在每次回发时刷新整个页面的需要，只需要更新 UpdatePanel 控件的内容即可。操作步骤如下。

（1）在项目中，打开“添加新项”窗口。在“模板”选项组中选择“AJAX Web 窗体”，在“名称”文本框中，将名称更改为“UseTimer.aspx”，然后单击“添加”按钮。

（2）如果页面尚未包含 ScriptManager 控件，则在工具箱的“AJAX Extensions”选项卡中双击 ScriptManager 控件，将其添加到页面中。然后再添加 UpdatePanel 控件。

（3）单击 UpdatePanel 控件使其成为被选中状态，然后双击 Timer 控件，则将其添加到 UpdatePanel 控件内。将 Timer 控件的 Interval 属性设置为 10000。Interval 属性是以毫秒为单位定义的。因此，若将 Interval 属性设置为 10000 毫秒，则会每 10 秒刷新一次 UpdatePanel 控件。

注意：Timer 控件可在 UpdatePanel 控件的内部或外部用作一个触发器。本示例演示了如何在 UpdatePanel 控件内部使用 Timer 控件。

（4）将一个 Label 控件添加到 UpdatePanel 控件内。修改其 Text 属性设置为“面板尚未刷新”，然后在 UpdatePanel 外部再添加一个 Label。

注意：在此示例中，计时器时间间隔设置为 10 秒。这样在运行示例时，读者无需等待很长时间就可以看到结果了。但是每个计时器时间间隔都会导致向服务器回发，从而引起网络通信，因此在成品应用程序中，应将此时间间隔设置为在应用中仍可执行的最长时间。

界面最终源代码如代码 13-03 所示。如果要使用 UpdatePanel 控件或 Timer 控件，则页面上必须添加 ScriptManager 控件。

代码 13-03　使用 Timer 控件

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>无标题页</title>
    <script type="text/javascript">
      function pageLoad( ) {
      }
    </script>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ScriptManager ID="ScriptManager1" runat="server" />
                    <!--下面是局部刷新内容-->
        <asp:UpdatePanel ID="UpdatePanel1" runat="server">
            <ContentTemplate>
```

```
                    <asp:Timer ID="Timer1" runat="server" Interval="10000" ontick="Timer1_Tick">
                    </asp:Timer>
                    <br />
                    <asp:Label ID="Label1" runat="server" Text="面板尚未刷新"></asp:Label>
                </ContentTemplate>
            </asp:UpdatePanel>
                    <!--以上是局部刷新内容-->
    </div>
    <asp:Label ID="Label2" runat="server"></asp:Label>
    </form>
</body>
</html>
```

后台功能代码如代码 13-04 所示主要是设置 Label2 和 Label1 标签的内容。

代码 13-04　使用 Timer 控件

```
public partial class UseTimer : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label2.Text = "页面创建于: " +
            DateTime.Now.ToLongTimeString( );//提示信息
    }
    protected void Timer1_Tick(object sender, EventArgs e)
    {
        Label1.Text = "面板刷新在: " +
          DateTime.Now.ToLongTimeString( );
    }
}
```

（5）在浏览器中查看 UseTimer.aspx，页面效果如图 13-5 所示。页面每隔 10 秒钟刷新一次，效果如图 13-6 所示。

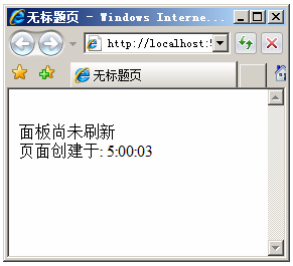

图 13-5　浏览 UseTimer.aspx 效果

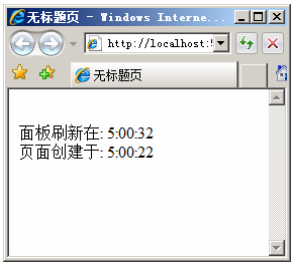

图 13-6　刷新效果

13.3　使用 Microsoft AJAX Library 创建自定义客户端脚本

ASP.NET AJAX 有助于创建客户端脚本，然后将其集成到 ASP.NET 应用程序中。这主要是通过 JavaScript 的类型系统、为 JavaScript 对象提供的类扩展等功能。ASP.NET 还包括 ScriptManager 控件，此控件可管理这些脚本库，以及应用程序中的任何自定义脚本。

13.3.1　AJAX Library 简介

Microsoft AJAX Library 能够实现的主要功能如下。

- 向 JavaScript 代码中添加面向对象的功能，用来提高代码的重用性、灵活性和可维护性。
- 在运行时用反射检查客户端脚本的结构。
- 用枚举提供另一种不同于整数的、易读的表示形式。
- 用 JavaScript 基类型的扩展，缩短脚本的开发时间。
- 使用调试扩展和跟踪功能，实现比传统 JavaScript 调试技术更快、信息更丰富的调试。

13.3.2　使用类型系统

Microsoft AJAX Library 增加了一个类型系统和一系列对 JavaScript 对象的扩展。利用这些功能，可按结构化方式编写支持 AJAX 的 ASP.NET 应用程序。这不仅能提高可维护性，还简化了添加功能。Microsoft AJAX Library 扩展为 JavaScript 添加了以下功能。

1．类、成员和命名空间

Microsoft AJAX Library 包括基类，还有一些由其派生的对象和组件。通过所有这些类，可以使用面向对象的编程模型来编写客户端脚本。

Type 类为 JavaScript 编程添加了命名空间、类和继承等面向对象的功能。使用 Type 类注册的 JavaScript 对象，都会自动获得访问这些功能的权限。如代码 13-05 所示，演示如何使用 Type 类，在 JavaScript 文件中创建并注册一个命名空间和类。

代码 13-05　创建并注册命名空间和类

```
Type.registerNamespace("Demo");                                    //注册命名空间
Demo.Person = function(firstName, lastName, emailAddress) {
    this._firstName = firstName;
    this._lastName = lastName;
    this._emailAddress = emailAddress;
}                                                                  //创建类
Demo.Person.prototype = {
    getFirstName: function( ) {
        return this._firstName;
    },
    getLastName: function( ) {
        return this._lastName;
    },
    getName: function( ) {                                         //Name 属性
        return this._firstName + ' ' + this._lastName;
    },
    dispose: function( ) {
        alert('bye ' + this.getName( ));
    }
}
Demo.Person.registerClass('Demo.Person', null, Sys.IDisposable);   //注册类
```

```
if (typeof(Sys) !== 'undefined') Sys.Application.notifyScriptLoaded( );
```

2．访问修饰符

访问修饰符指定类或成员可用的上下文。JavaScript 中虽然没有访问修饰符，但是 Microsoft AJAX Library 遵循约定：名称是以下划线字符（“_”）开头的成员，是类的私有成员，不能从类的外部访问它们。

3．继承

继承是指一个类从另一个类派生的能力。派生类可自动继承父类所有的字段、属性、方法和事件。派生类还可以自由添加新成员或重写基类的现有成员，以拓展这些成员的行为。

4．接口

接口用于定义实现类的输入和输出要求。这样函数可以和实现同一接口的类进行交互，而不考虑该类还实现了哪些其他功能。

13.4 应用 AJAX 工具包

通过前面的学习可以知道，开发人员还可以利用 AJAX 提供的 JS 库，扩展或者自定义一些控件。ASP.NET 网站上提供了很多好的这些控件，都是扩展并封装好的。本节就介绍如何下载并使用它们。

13.4.1 下载并安装 AJAX 工具包

AJAX 工具包的下载地址是“http://www.asp.net/ajax/ajaxcontroltoolkit/”，下载界面如图 13-7 所示。单击图 13-7 中标记的按钮，会打开一个选择下载包类型的界面，如图 13-8 所示。这里包含带源代码的包和不带源代码的包，读者根据喜好选择一个即可。为了下载方便，本例选择了第 2 项，只带一个 DLL 的下载包。

下载后，只有一个 bin 文件夹，其中有一个 AjaxControlToolkit.dll 文件，这就是封装好的组件库。现在来学习如何添加该组件库到工具箱中。

（1）右击工具箱的空白处，在弹出的快捷菜单中单击“添加选项卡”命令。新选项卡命名为 AJAXTOOL。

（2）右击新选项卡 AJAXTOOL，在弹出的快捷菜单中单击“选择项”命令，打开“选择工具箱项”窗口，如图 13-9 所示。

（3）单击“浏览”按钮，找到刚刚下载的 dll 文件，选中后单击“打开”按钮，此时选择工具箱的效果如图 13-10 所示，其中所选择的这些项就是 dll 中包含的所有组件。

（4）单击“确定”按钮，回到工具箱。此时选项卡 AJAXTOOL 下的内容如图 13-11 所示，这里包括 40 多个常用 AJAX 组件。

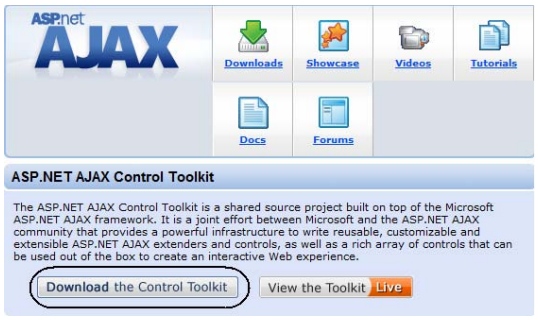

图 13-7　下载界面

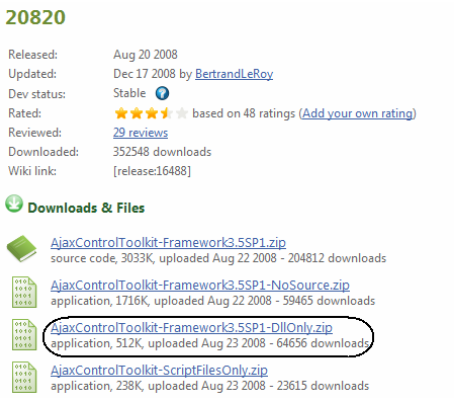

图 13-8　选择下载包类型

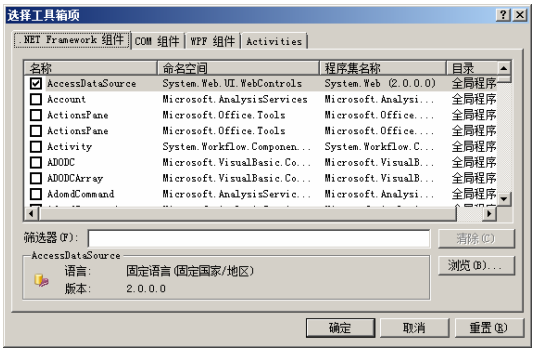

图 13-9　“选择工具箱项”窗口

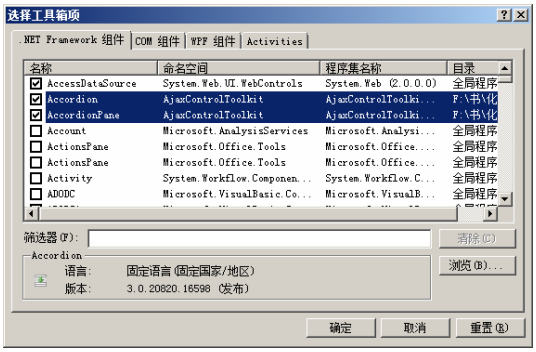

图 13-10　选中的组件

注意：不同时期下载的 AJAX 组件数量不同，因为微软公司在不断地添加组件。

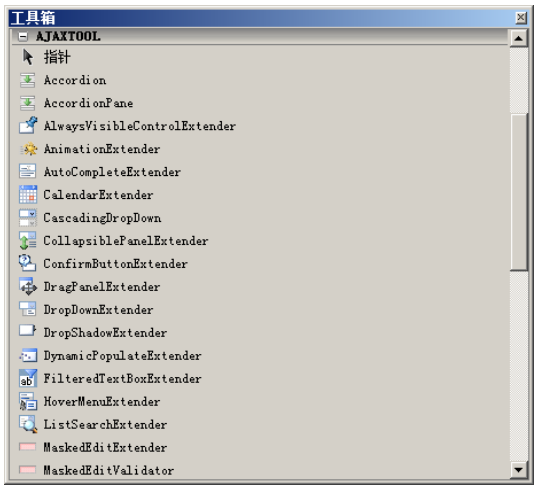

图 13-11　选项卡 AJAXTOOL 中的控件

13.4.2 实现文本框水印效果

在新添加的控件中，可以看到一个 TextBoxWatermarkExtender 控件。顾名思义，这个控件实现的是文本框水印效果。水印就是用户还没有填写文本框时，文本框内给出一些提示文字，用户把光标放到文本框上时，这些提示自动消失。本节通过一个示例来介绍它的使用方法。

（1）在当前项目中添加一个 Web 窗体，命名为“UserWater.aspx”。

（2）拖动一个 ScriptManager 到设计界面，然后设计界面为一个登录页面，如图 13-12 所示。

（3）注意，TextBoxWatermarkExtender 控件不是直接拖到设计页面，而是单击用户名后面的 TextBox1，出现如图 13-13 所示的提示。

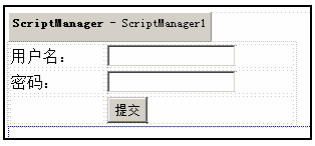

图 13-12　设计界面

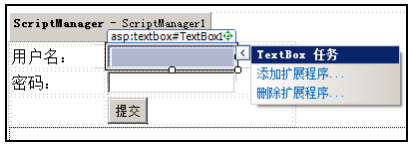

图 13-13　TextBox 的扩展选项

（4）选择“添加扩展程序”选项，就打开了“扩展程序向导”窗口，如图 13-14 所示。这里显示了所有支持 TextBox 的组件，找到 TextBoxWatermarkExtender，单击“确定”按钮。

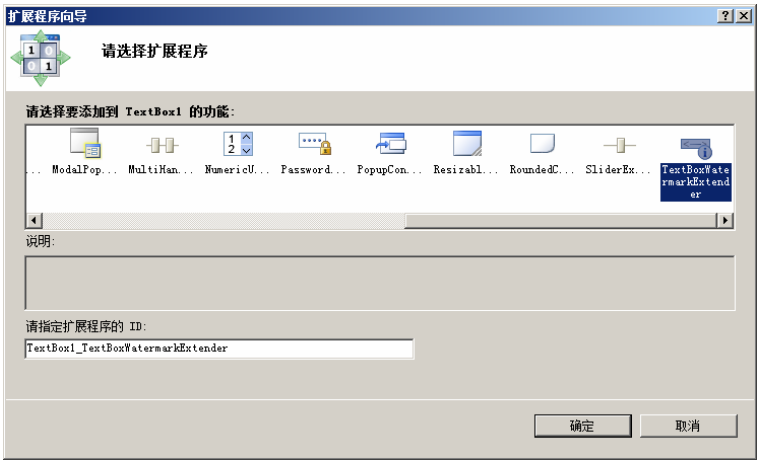

图 13-14　“扩展程序向导”窗口

（5）同样针对 TextBox2 进行操作。然后打开源代码视图，设计两个 TextBoxWatermarkExtender 控件的属性，让其实现水印提示效果，如代码 13-06 所示。

代码 13-06　TextBoxWatermarkExtender 的设计

```
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<cc1:TextBoxWatermarkExtender ID="TextBox1_TextBoxWatermarkExtender"
                              runat="server" Enabled="True"
                              TargetControlID="TextBox1"
```

```
                        WatermarkText="请输入姓名"
                        WatermarkCssClass="water1">
</cc1:TextBoxWatermarkExtender>
   <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
<cc1:TextBoxWatermarkExtender ID="TextBox2_TextBoxWatermarkExtender"
                        runat="server" Enabled="True"
                        TargetControlID="TextBox2"
                        WatermarkText="请输入密码"
                        WatermarkCssClass="water2">
</cc1:TextBoxWatermarkExtender>
```

水印控件主要有以下 3 个属性。

- TargetControlID：应用水印效果的文本框控件 ID。
- WatermarkText：水印效果显示的文本。
- WatermarkCssClass：水印效果应用的样式表。

本例应用了样式 water1，其内容如代码 13-07 所示。

代码 13-07　样式表设计

```
<style>
      .water1 {                                  <!--水印样式 1-->
       height:20px;
       width:150px;
       padding:2px 0 0 2px;
       border:1px solid  #BEBEBE;
       background-color:#F0F8FF;
       color:gray;}
      .water2 {                                  <!--水印样式 2-->
       height:20px;
       width:150px;
       border:1px none  #ffffcc;
       background-color:#ccccff;
       color:Green;}
</style>
```

（6）在浏览器中查看当前页面，效果如图 13-15 所示。

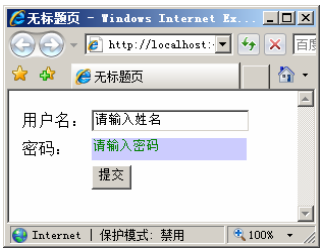

图 13-15　浏览效果

13.5　小结

ASP.NET AJAX 可以让开发者发挥出浏览器中 Web 应用程序处理最出色的一面，而不需要跟服务器端交互来更新页面。ASP.NET AJAX 开发集成了 ECMAScript（JavaScript）客户端脚本库和 ASP.NET 3.5 基于服务器端的开发平台。ASP.NET AJAX 依赖于 AJAX 策略来创建 Web 应用程序，这样就能通过客户端脚本向基于 Web 的应用程序发送请求。本章各种 AJAX 控件的示例都很简单，读者需要在实际项目中更多地动手应用，才能完全掌握 AJAX 技术。