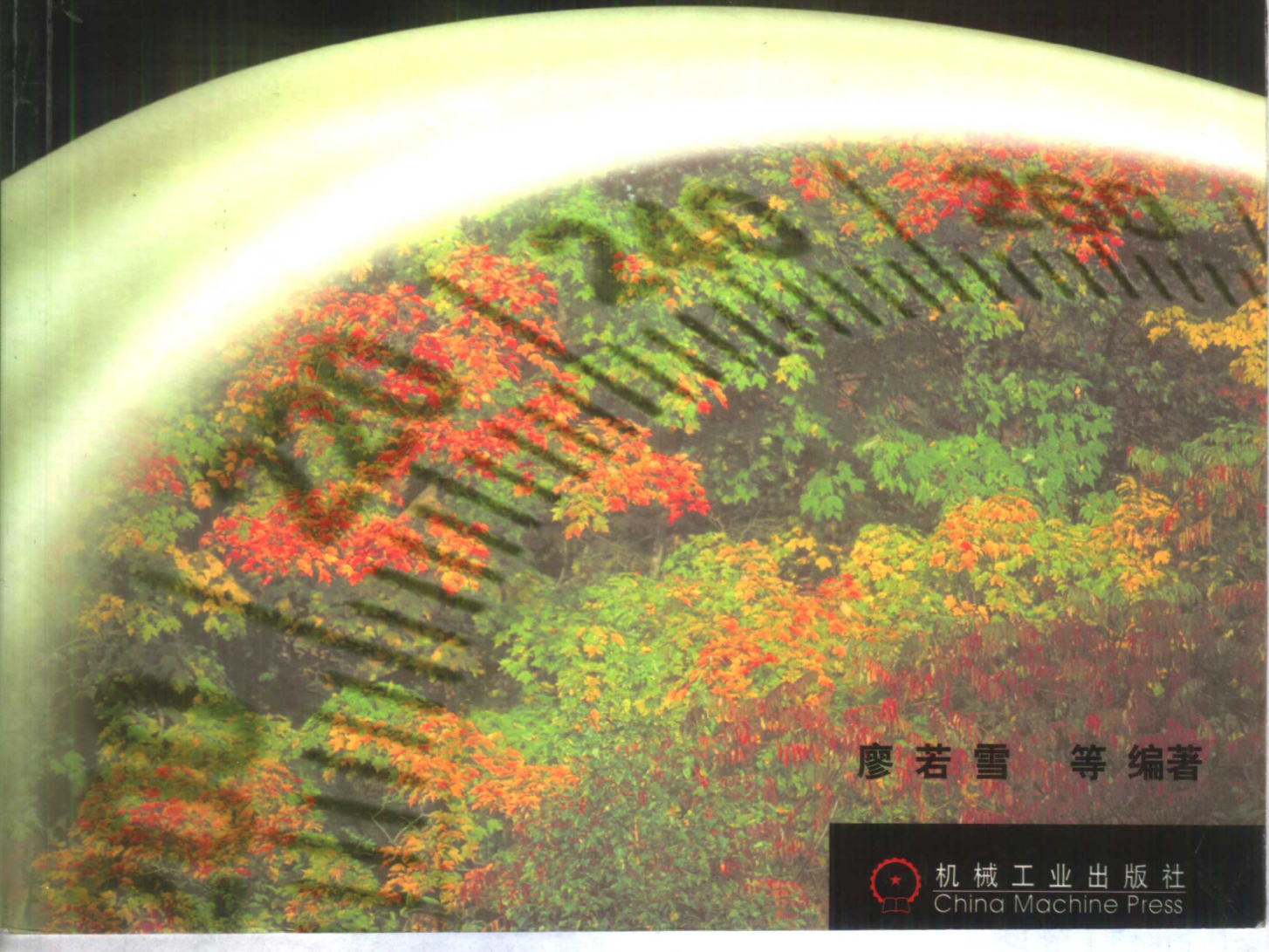
HZ BOOKS

软件开发技术丛书

ASP.Net

动态网站编程指南



廖若雪 等编著

机械工业出版社
China Machine Press

软件开发技术丛书

ASP.Net动态网站编程指南

廖若雪　等编著

机械工业出版社
China Machine Press

本书介绍ASP.Net动态网站开发技术。主要内容包括ASP.Net Web窗体和服务器端组件，ASP.Net应用程序和指示，数据的读取、显示和操作，使用Pagelet，ASP.Net使用技巧等。展示了ASP.Net这种全新的动态网站开发工具的全面功能。

本书适合有传统动态网站开发技术基础的读者阅读。

图书在版编目（CIP）数据

ASP.Net动态网站编程指南/廖若雪等编著. -北京：机械工业出版社，2001.7
（软件开发技术丛书）
ISBN 7-111-09046-2

Ⅰ.A… Ⅱ.廖… Ⅲ.主页制作-应用软件，ASP.Net Ⅳ.TP393.092

中国版本图书馆CIP数据核字（2001）第039517号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）
责任编辑：谢 昱
北京第二外国语学院印刷厂印刷 · 新华书店北京发行所发行
2001年7月第1版第1次印刷
787mm × 1092mm1/16 · 25.5印张
印数：0 001-5 000册
定价：39.00元

前　言

本书研究了微软公司最新的互联网技术——ASP.Net，讲解了ASP.Net技术在动态网站开发方面的应用。

本书的读者

本书主要面向那些希望尽快学会使用ASP.Net技术的读者，需要读者具有基本的互联网和动态网站开发的概念。

ASP.Net是一种完全基于组件的技术，所以读者如果有组件技术方面的知识，那么会对学习本书有相当的帮助。

如何使用本书

ASP.Net是一门全新的技术，学习这门新的技术，理解技术的本质比掌握技术本身更为重要。所以，本书的学习主要可以分为两个阶段：

1) 理解技术实质。

2) 学习具体的技术。

在第一个阶段，建议读者快速浏览本书，通过大量的代码示例理解ASP.Net动态网站开发和传统的动态网站开发之间的异同。特别是对于已经有动态网站开发经验的读者，需要彻底地抛弃原有旧的思维模式，学习按照ASP.Net的思想去运用ASP.Net。

第二个阶段和传统的学习方式差不多，需要仔细地学习书中的每一个方面，配合文档资料和大量的练习，彻底掌握ASP.Net这门最新的技术。

到目前为止，ASP.Net仍然属于一种“开发中”的技术，所以作者在这里不保证正式发布的ASP.Net和本书讲解的内容完全一致，不过，对于Visual Studio.NET beta1和beta2，本书的内容将是完全可信的。

ASP.Net是一门非常激动人心的新技术，希望本书能够帮助想学习这门新技术的读者尽快地掌握ASP.Net。

本书由廖若雪主持编写，参加本书编写工作的还有：张卉、赵仁杰、徐江红、卿理、肖扬、顾超、张柯、马飞、江庆铭、陆箐、张泽湘、刘伟、陈威志、李辉、郑为、宋鹏、唐绍南、余峰、徐科、杨君守、杨华、李毅、陈蕊、胡建伟、邓青等。

由于编著水平有限，疏漏之处在所难免，请读者指正。

廖若雪

2001年1月

目　录

前言

第1章　概述 ……1

1.1　动态网站开发技术的发展……1

1.2　ASP.Net概述 ……1

1.3　实验平台的搭建……1

1.4　预备知识……4

1.4.1　C#语言 ……4

1.4.2　回调函数 ……4

1.4.3　ASP.Net动态网站开发技术的特点……4

1.5　本书的学习方法……5

第2章　ASP.Net Web窗体和服务器端组件 ……7

2.1　Web窗体功能概述 ……7

2.1.1　第一个ASP.Net页面……7

2.1.2　ASP.Net中的代码段……8

2.1.3　ASP.Net服务器端组件概述 ……11

2.1.4　自定义服务器端组件……14

2.1.5　数据显示组件概述……22

2.1.6　数据验证……29

2.1.7　分离代码和页面……33

2.1.8　小结……35

2.2　ASP.Net服务器端组件使用初步……36

2.2.1　组件的声明和属性设定……36

2.2.2　组件的标识……39

2.2.3　组件事件的处理……41

2.2.4　页面定向……45

2.3　ASP.Net服务器端组件的样式……51

2.3.1　利用ASP.Net动态改变HTML元素的样式……51

2.3.2　设置ASP.Net服务器端组件的样式 ……58

2.3.3　样式设置的综合使用……68

2.4　基本的服务器端组件 ……75

2.4.1　Label ……75

2.4.2　TextBox ……76

2.4.3　Button、LinkButton和ImageButton ……77

2.4.4　CheckBox和CheckBoxList……78

2.4.5　RadioButton和RadioButtonList ……80

2.4.6　Image ……80

2.4.7　HyperLink ……81

2.4.8　Table、TableCell和TableRow ……83

2.4.9　DropDownList ……84

2.5　服务器端组件数据验证的使用 ……86

2.5.1　数据验证概述……86

2.5.2　服务器端数据验证和客户端数据验证…91

2.5.3　显示验证失败信息……96

2.5.4　使用比较数据验证组件 ……100

2.5.5　使用范围数据验证组件 ……104

2.5.6　正则表达式 ……107

2.5.7　使用正则表达式数据验证组件 ……110

2.5.8　自定义数据验证组件的使用 ……116

2.5.9　综合应用 ……118

第3章　ASP.Net应用程序和指示 ……126

3.1　Global.asx文件 ……126

3.2　ASP.Net指示 ……128

3.2.1　Page ……128

3.2.2　Import ……128

3.2.3　Assembly ……129

第4章　数据的读取、显示和操作 ……130

4.1　数据绑定……130

4.1.1　基本的数据绑定 ……130

4.1.2　基于变量的数据绑定 ……131

4.1.3　ASP.Net服务器端组件的数据绑定……133

4.1.4　ASP.Net服务器端组件作为数据源……135

4.1.5　在数据绑定中使用函数和表达式 ……139

4.1.6　使用二维数据源 ……141

4.1.7 列表绑定组件 ……147
4.2 数据库操作基础：ADO+ ……149
4.2.1 数据库连接的建立 ……149
4.2.2 数据库查询 ……150
4.2.3 DataSet组件的使用 ……153
4.3 数据库和XML数据的访问 ……156
4.3.1 数据显示组件DataGrid的使用 ……156
4.3.2 数据添加的实现 ……160
4.3.3 为数据的添加提供数据验证服务 ……165
4.3.4 实例：一个简单的数据库数据管理程序 ……172
4.3.5 存储过程的使用 ……211
4.3.6 使用XML文件作为数据源 ……216
4.4 自定义数据显示 ……226
4.4.1 DataList组件概述 ……226
4.4.2 使用Repeater组件 ……227
4.4.3 使用DataList组件 ……231
第5章 使用Pagelet ……240
5.1 最简单的Pagelet ……240
5.2 包含属性的Pagelet ……241
5.3 数据库查询的Pagelet ……250
5.4 动态载入Pagelet的ASP.Net服务器端组件 ……253
第6章 ASP.Net使用技巧 ……257
6.1 文件处理 ……257
6.2 文件上传 ……258
6.3 邮件发送 ……258
6.4 ASP.Net的调试 ……259
附录A 预定义服务器端组件 ……263
附录B C#语言 ……344

第1章 概 述

1.1 动态网站开发技术的发展

动态网站开发技术到目前为止经历了四代：

1) CGI时代。那是使用C、Perl、VB开发动态网站的时代，动态网站的开发在当时仅仅只有少数程序设计人员掌握。

2) -SAPI时代。代表是NSAPI和ISAPI，从开发者的角度讲，这种开发方式并没有带来开发上的方便。

3) 脚本语言时代。这时期涌现出许多杰出的脚本语言：ASP、PHP、嵌入式Perl和JSP。脚本语言的出现大大简化了动态网站开发的难度，特别是ASP和PHP，学习简单、功能强大，成为许多网站开发的首选。

4) 组件技术时代。ASP.Net和J2EE技术是这个时代的代表，尽管本书是讲解ASP.Net的，但是J2EE技术的强大也是显而易见的。

1.2 ASP.Net概述

ASP.Net是编译执行的Web服务器端开发工具，和JSP一样，ASP.Net也是预先编译为一个类文件，当用户访问ASP.Net文件的时候，直接执行这个类文件而不是原先的类文件，在编译前可能和JSP一样，将ASP.Net源文件翻译成一个C#或VB的源代码文件。

读者也许还听说过“ASP+”这种说法，实际上，ASP.Net就是ASP+，由于“ASP.Net”这种说法使用得比较多，所以本书使用“ASP.Net”。

1.3 实验平台的搭建

对ASP比较熟悉的读者都知道，ASP页面实际上是由一个名为asp.dll的ISAPI动态链接库解释执行的，实际上，ASP.Net页面也需要有一个ISAPI动态链接库来解释、处理，这个动态链接库当前的名称为xspisapi.dll，那么是不是只需要这样一个动态链接库就可以搭建一个ASP.Net的实验平台了呢？当然不行，这个动态链接库还需要自己的软件运行环境，这就是NGWS Runtime。

对于仅仅需要学习ASP.Net的读者，没有必要去深入地了解NGWS（Next Generation Windows Services，下一代窗口服务）所以，本书将关于NGWS的信息放在了附录B中。

NGWS Runtime和SDK可以从微软公司的Web站点免费下载，由于NGWS计划在Visual Studio 7中出正式版本，所以现在能下载的是NGWS Runtime和SDK的测试版，具体版本号是2000.14.1812。

在下载这个80多兆字节的大文件之前，首先需要检查一下自己的计算机是否能够运行NGWS，在Microsoft的网站上提出的要求如下：

• CPU: Intel Pentium II-class 300 MHz (Intel Pentium III-class 600 MHz recommended)

说明：这表明需要至少300MHz的Pentium II级别的处理器，笔者使用的是Celetron 433，运行速度还可以。

• RAM: 96 MB (128 MB recommended)

说明：128M的内存实际运行起来还是觉得不够，最好能够有256M内存。

• Available hard disk space (for install): 250 MB

说明：鉴于现在硬盘的价格，读者应该很容易就实现这个要求。

• Available hard disk space (post install): 155 MB

说明：鉴于现在硬盘的价格，读者应该很容易就实现这个要求。

• Video: 800 × 600, 256 colors

说明：14英寸显示器就可以达到要求。

• CD-ROM: required

说明：目前计算机的标准配置，不过，笔者没有发现在NGWS的安装中CD-ROM有什么用。

• Operating System: Microsoft Windows 2000 and Microsoft Internet Explorer 5.5

说明：使用Windows 98和Windows NT的读者需要升级自己的系统到Windows 2000以安装NGWS。至于Microsoft Internet Explorer 5.5，在笔者看来，如果读者仅仅使用ASP.Net，则没有必要安装Microsoft Internet Explorer 5.5。不过由于在NGWS安装的开始需要检查Internet Explorer的版本号，所以读者可以手动地在注册表里将Internet Explorer的版本号提高，这里就不提供具体的方法了。

• Other Software: MDAC 2.6 Beta 2

说明：符合以上条件就不需要考虑这个问题了。

最后，有一点是Microsoft没有说明的，为了提供Web服务，需要有Internet Information Server，Windows 2000的Server和Advanced Server自带了Internet Information Server，如果读者使用的是Professional版本的Windows 2000，就需要手动从Server版本的Windows 2000光盘中添加组件至系统中。

好了，如果以上的条件都得到了满足，就可以下载这个80多兆字节的庞然大物了，具体下载地址是：

http://download.microsoft.com/download/platformsdk/Trial/1812.10full/NT5/EN-US/Setup.exe

下载完后，安装及配置很简单，只需要运行这个setup.exe，然后按照向导完成安装即可。

完成安装以后，就可以像使用一般的HTML和ASP文件一样使用ASP.Net文件了。

下面是一个测试用的ASP.Net文件，来看一看系统是否已经正常工作了：

```
<%@ Page Language="C#" %>
<HTML>
    <HEAD>
        <TITLE>
            第一个ASP.Net文件，测试用。
        </TITLE>
    </HEAD>
```

```
    <BODY>
        下面使用ASP.Net输出一行文字：<BR>
        <%="第一个ASP.Net文件"%>
    </BODY>
</HTML>
```

将这个文件放在一个Web发布目录中，然后使用Web浏览器访问，读者也许会发现自己看见的页面中出现了一大堆“?”号，这可以用下面的方法解决：

1) 找到文件winnt/complus/[version]/config.web，这里的[version]指的是NGWS的版本号，例如winnt/complus/2000.14.1812/config.web。

2) 使用文本编辑器，例如NotePad或UltraEdit，找到下面几行：

```
<globalization
    requestencoding="us-ascii"
    responseencoding="iso-8859-1"
/>
```

很可能在这个文件的29~32行。

3) 修改为下面几行：

```
<globalization
        requestencoding="gb2312"
        responseencoding="gb2312"
/>
```

其实，只需要修改responseencoding属性就可以让中文的显示正常了，但是，如果不修改requestencoding属性的值，会出现其他问题。

现在应该可以了，具体的页面效果如图1-1所示。

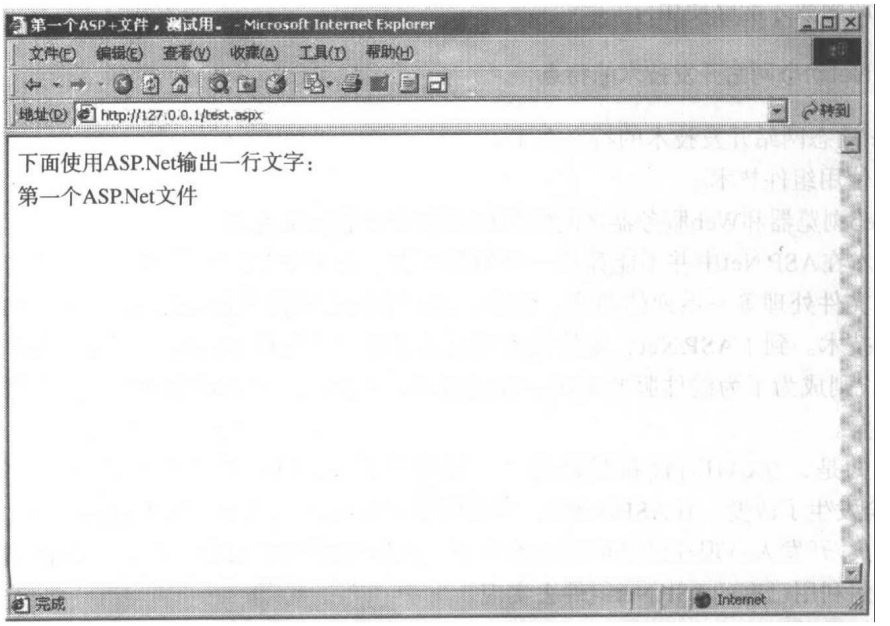


图 1-1

1.4 预备知识

1.4.1 C#语言

C#语言是微软公司专门为NGWS而开发的新一代程序设计语言，C#和C/C++以及Java语言非常相似，C/C++或Java程序员仅需要很少的时间就可以掌握C#。

ASP.Net的开发中尽管可以使用多种开发语言，本书中的例子却都是使用C#的，原因主要有两个：

1) C#比VB更加适合于开发基于组件的程序。

2) 没有必要为每个例子提供两种版本的代码。

实际上，由于ASP.Net本身是建立在NGWS组件库之上的，所以使用何种语言开发ASP.Net并不是一件重要的事情，在大部分情况下，根本就不需要使用C#或者VB的特性去开发ASP.Net程序。

本书不打算详细地讲解C#，也不打算介绍VB。不过由于C#是一种比较新的语言，考虑到大多数的读者还没有使用C#的经验，所以本书为读者提供了一个C#的基础教程，放在附录B中。读者可以自行学习。

1.4.2 回调函数

在利用ASP.Net组件开发动态网站的时候，会常常遇到所谓的“回调函数”，所谓“回调函数”，就是用户定义一个函数（或者称为方法），将函数名传递给一个ASP.Net组件，然后这个组件就可以在需要的时候调用这个函数来完成用户要求的任务。

需要注意的是，ASP.Net中的回调函数都有一定的声明形式，只有声明形式符合ASP.Net控件要求的回调函数才能够使用。

1.4.3 ASP.Net动态网站开发技术的特点

ASP.Net动态网站开发技术的特点在于：

1) 大量使用组件技术。

2) 将Web浏览器和Web服务器之间的网络通信完全地包装起来。

组件技术在ASP.Net中并不能算是一个新的概念，在ASP中，就利用组件技术实现了诸如数据库连接、文件处理等一系列的功能，但是，ASP中核心的技术还是JavaScript、VBScript这样的脚本语言技术。到了ASP.Net，组件技术则成为了整个开发技术的核心，而作为程序语言出现的C#和VB，则成为了为组件服务的次一级的技术，实际上，在ASP.Net中，使用哪一种开发语言根本不重要。

更重要的是，在CGI时代就开始的“利用程序输出HTML代码”的动态网站开发技术到ASP.Net完全发生了改变。在ASP.Net中，动态网站开发技术已经变为利用组件搭建“基于Web的应用程序”了。开发人员现在已经可以基本上不考虑如何使用HTML代码去实现自己需要的效果，而是考虑如何利用已有的ASP.Net组件去实现。

例如，如果需要构建一个电子商务网站的购物系统，那么需要如下一些组件：

ADO+中的数据库连接组件。

DataSet数据集组件。

DataGrid数据显示组件。

一组数据验证组件。

利用这些组件构成的ASP.Net页面的结构如图1-2所示。

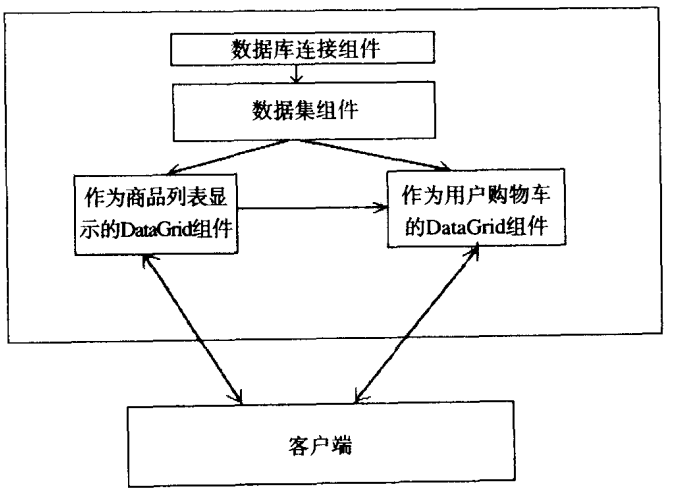


图 1-2

尽管读者现在可能还不能理解这些组件的具体作用和使用方法，不过仍然可以理解整个ASP.Net页面的工作机制：

数据库连接组件负责处理数据库连接。

数据集组件相当于数据库的一个抽象，可以通过数据库连接组件完成数据查询、删除、更新、添加等数据操作。

DataGrid组件显示数据和更新数据，当然这一切都是通过数据集组件完成的。

当客户端访问这个ASP.Net页面的时候，首先得到的是由显示商品列表的DataGrid组件通过数据集组件经过数据库连接组件从数据库中读出的商品列表，当用户要求购买某一件商品的时候，则通过作为用户购物车的DataGrid组件通过数据集组件经过数据库连接组件向数据库中添加一条购物信息。

就这么简单地使用了几个ASP.Net服务器端组件就完成了一个B2B的电子商务站点，这就是组件技术的威力所在。

在ASP.Net中，开发人员根本就不用关心服务器和客户端之间的数据传递，就像开发普通的应用程序一样，可以直接将某个组件的事件和一个自定义的函数（或称为方法）关联起来，尽管这个事件是从客户端产生的，但是开发人员却可以完全不考虑这些。

1.5 本书的学习方法

ASP.Net是一种全新的动态网站开发工具，ASP.Net的学习方法也和学习传统的动态网站开

发技术有很大的差别。笔者就曾经试图使用学习ASP、PHP、JSP的方法去学习ASP.Net，结果走了很多弯路，为使读者能够尽快地掌握这一门崭新的开发技术，这里讲解一下本书的学习方法。

首先，需要明确，最重要的是组件的使用而不是程序代码的编写，这是和传统开发技术的最大差别。从本书的目录就可以看出，本书大部分的内容都是在讲解如何使用ASP.Net服务器端组件，读者应该知道，掌握这些组件的用法并能够灵活应用这些组件是成为ASP.Net专家的第一步。

其次，要避免使用传统的程序设计方法。这不是绝对的，但是，能够使用ASP.Net的组件技术解决的问题最好不要使用传统的程序设计方法去做，这样做的目的一是为了改变自己的开发观念和开发思维，二是因为ASP.Net服务器端组件之间很容易发生交互，而自己写的代码可能就没有那么容易了。

最后，为不同的读者给出学习本书的一些建议。

对于学习过Java技术的读者，如果已经有C#的基础，那么可以从本书的第1章开始，一直看到最后一章。如果没有C#基础，那么需要预先浏览一下附录B中的C#教程。

对于一直使用传统的动态网站开发技术的读者，无论有没有C#基础，都应该首先对NGWS有所了解，不需要太多，但是一定要有面向组件程序开发的概念。

对于没有任何基础的读者，那么本书对您可能有些难度，不过不要紧，没有传统程序设计的思维定式也是您的一大优势。这部分读者需要首先了解一些HTML和程序设计方面的知识，本书不是一本HTML方面的书，所以没有讲解任何HTML的知识，读者可以从另外一些书籍中找到HTML的知识，程序设计的知识可以从本书附录B的C#教程中得到，不过需要更多的努力。

最后，如果使用Visual C++、Delphi、C++Builder等的传统桌面应用程序的开发人员也对本书感兴趣，那么不需要笔者多说，以这类读者的知识和经验，阅读本书应该是轻而易举的。

无论哪一种类型的读者，在学习过程中如果有什么不明白的，都可以参考附录A和附录B，笔者相信大部分学习当中的问题可以在附录中找到答案。

第2章 ASP.Net Web窗体和服务器端组件

和以往的动态Web开发工具不一样，ASP.Net并不是一种传统意义上的替代CGI的工具，从某些方面来说，ASP.Net的开发更像Windows桌面应用程序的开发。在ASP.Net中，开发人员无需关心浏览器和服务器的区别，也无需手动处理客户端发送到服务器端的数据，开发人员完全可以把精力放到实现网站的显示效果和网站的功能上。

在ASP.Net中，实现这些功能的主要模块称为ASP.Net Web窗体（Form），本章将详细讨论ASP.Net Web窗体，以及各种各样的ASP.Net服务器端组件。

2.1 Web窗体功能概述

ASP.Net Web窗体框架结构是一种可伸缩的NGWS运行期程序模块，这个模块主要用于在服务器端动态地创建Web页面，也就是说，有了这个模块，开发人员也就不需要手动地创建动态Web页面了。

ASP开发人员会发现，ASP.Net的Web窗体包括了以前的ASP的全部功能。作为ASP的扩展，ASP.Net也支持原有的ASP语法，也就是说，在ASP.Net Web窗体中可以使用原有的ASP语法。

相对于ASP，ASP.Net的Web窗体具有以下的特点：

1) 开发人员可以轻松地建立可重用的用户界面控件。

2) 开发人员可以简洁地建立网站页面逻辑。

3) 开发工具提供了真正的所见即所得的开发环境。

在本节中，将概述性地介绍一下ASP.Net的Web窗体，读者不需要完全理解这一节中的每个例子，但是，需要读者理解每个例子所表达的ASP.Net概念和ASP.Net功能。特别是对于使用过ASP、PHP等传统的服务器端脚本技术的读者，需要在本节中改变传统的观念，建立新的服务器端编程概念。

2.1.1 第一个ASP.Net页面

ASP.Net页面实际上仅仅是一个文本文件，不过，这个文件具有aspx的后缀名，当用户通过浏览器透过服务器访问这个页面的时候，NGWS将分析和编译这个aspx文件，生成一个称为NGWS类的二进制文件，这个二进制文件会自动地随着aspx源文件的更新而自动更新，而当用户访问这个页面的时候，实际上是这个NGWS类二进制文件在运行、处理用户的请求。对JSP熟悉的读者马上就会发现，这实际上也是JSP的运行模式。

尽管本书在第1章已经给出了一个可以运行的ASP.Net测试文件，但是，笔者觉得，那里并没有要求读者理解那个文件，所以，将下面的文件称为本书的第一个ASP.Net页面：

```
<html>
    <head>
```

```
<title>
            第一个ASP.Net文件
</title>
</head>
<body>
    <form action="1.aspx" method="post">
        <h3> 名称: <input id="Name" type=text>
        种类:  <select id="Category" size=1>
                    <option>物理学</option>
                    <option>化学</option>
                    <option>数学</option>
                </select>
        <input type=submit value="查找">
    </form>
</body>
</html>
```

读者也许会马上说：这不是一个HTML文件吗？不错，的确如此，但是如果将这个文件存为一个ASP.Net文件，也就具有了ASP.Net文件的基本特征了。

通过浏览器透过服务器访问这个文件，会发现页面的显示有着相当的滞后，这就是NGWS在编译这个ASP.Net文件了，页面的显示效果如图2-1所示。

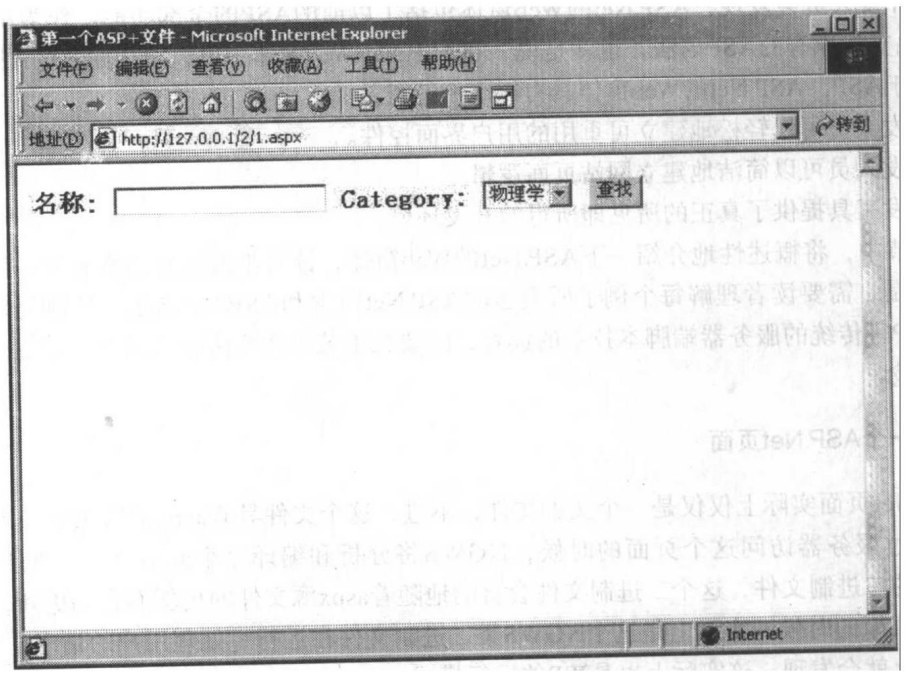


图 2-1

2.1.2 ASP.Net中的代码段

同ASP一样，ASP.Net中也可以使用<%...%>来区别ASP.Net代码和HTML页面代码，当然，

这种“相似”仅仅是表面上的。

下面的代码展示了如何在<%...%>中使用ASP.Net的流程控制语句来循环的输出字符串：

```
<%@ Page Language="C#"%>
<html>
    <head>
     <title>欢迎来到ASP.Net的世界！</title>
    </head>
    <body>
        <form action="1.aspx" method="post">
            <h3> 名称：<input id="Name" type=text>
            种类：<select id="Category" size=1>
                        <option>物理学</option>
                        <option>化学</option>
                        <option>数学</option>
                    </select>
            <input type=submit value="查找">
            <% for (int i=0; i <8; i++) { %>
                <font size="<%=i%>"> 欢迎来到ASP.Net的世界！</font> <br>
            <% }%>
        </form>
    </body>
</html>
```

显示效果如图2-2所示。

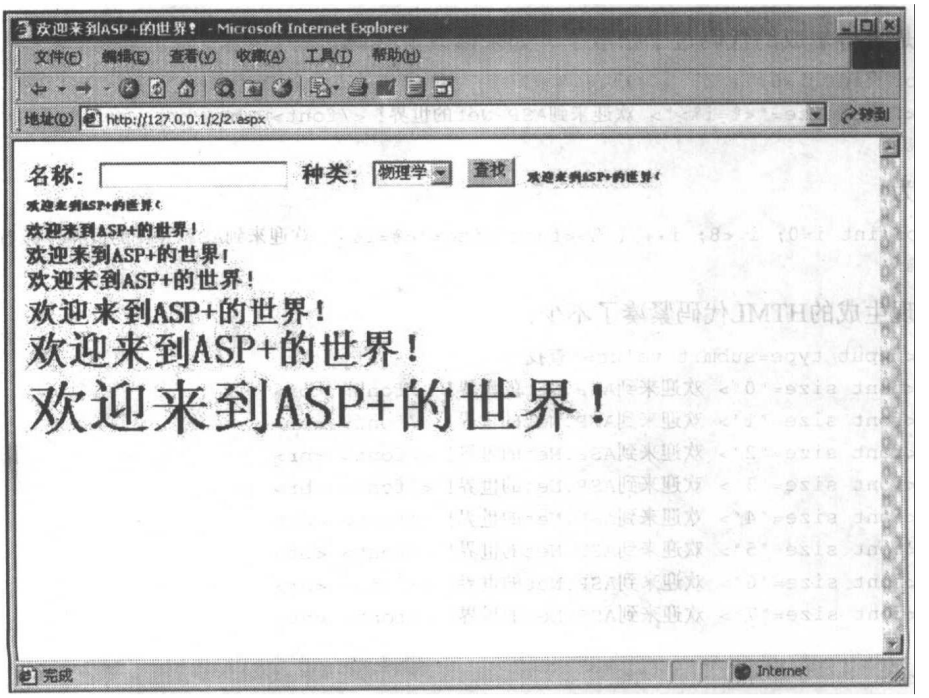


图 2-2

C#的语法在附录B中有介绍，这里就不说了，上面的代码很简单，就是循环输出“欢迎来到ASP.Net的世界！”这一串字符。现在来看一个比较有意思的现象，下面是上面代码生成的HTML文件的部分代码。

```
......
    <input type=submit value="查找">

        <font size="0"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="1"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="2"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="3"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="4"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="5"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="6"> 欢迎来到ASP.Net的世界！ </font> <br>

        <font size="7"> 欢迎来到ASP.Net的世界！ </font> <br>

        </form>
......
```

是不是觉得生成的代码过于松散了，如果修改源代码，将：

```
<% for (int i=0; i <8; i++) { %>
    <font size="<%=i%>"> 欢迎来到ASP.Net的世界！ </font> <br>
<% }%>
```

修改为：

```
<% for(int i=0; i <8; i++){ %><font size="<%=i%>">欢迎来到ASP.Net的世界！ </font> <br>
<% }%>
```

会发现生成的HTML代码紧凑了不少：

```
    <input type=submit value="查找">
    <font size="0"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="1"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="2"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="3"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="4"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="5"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="6"> 欢迎来到ASP.Net的世界！ </font> <br>
    <font size="7"> 欢迎来到ASP.Net的世界！ </font> <br>

</form>
```

这里可以总结出一条规律：ASP.Net代码段如果没有输出字符，那么也会输出一个空行。

部分HTML元素对空行和回车比较敏感，读者应该注意这个问题。

2.1.3 ASP.Net服务器端组件概述

在上一节中，展示了在ASP.Net中使用传统的动态网站开发办法，在本节中，读者将会看到一种崭新的动态网站开发模式，那就是ASP.Net服务器端组件。

这里解释一下什么是“ASP.Net服务器端组件”。首先，从语法上看，一个“ASP.Net服务器端组件”带有一个“runat="server"”的属性。其次，几乎每个“ASP.Net服务器端组件”都可以对应Web页面上的元素，例如：

HTML表单	对应	<form runat=server>
HTML文本框	对应	<asp:textbox runat=server>
HTML下拉列表框	对应	<asp:dropdownlist runat=server>
HTML按钮	对应	<asp:button runat=server>

而“ASP.Net服务器端组件”最重要的特点则是：开发人员在使用ASP.Net服务器端组件开发Web应用的时候可以像开发普通单机应用程序一样，一切和网络有关的概念都被隐藏到了这些“ASP.Net服务器端组件”中。

在ASP.Net中，对传统的HTML元素的操作也可以通过ASP.Net代码实现，只需要给任意一个HTML元素一个id值，然后在其属性中设置runat="server"，就可以在ASP.Net代码中通过这个id值控制这个HTML元素了。

下面的例子使用了上面列出的四个“ASP.Net服务器端组件”，首先来看一看代码：

```
<%@ Page Language="C#"%>
<html>
    <head>
    <title>
            ASP.Net的服务器端控制
        </title>
    </head>
    <body>
        <form action="3.aspx" method="post" runat=server>
            <h3> 名称: <asp:textbox id="Name" runat="server"/>
            种类:  <asp:dropdownlist id="Category" runat=server>
                        <asp:listitem>物理学</asp:listitem>
                        <asp:listitem>化学</asp:listitem>
                        <asp:listitem>数学</asp:listitem>
                    </asp:dropdownlist>
              <asp:button type=submit text="查找" OnClick="SubmitBtn_Click" runat=
"server"/>
        </form>
    </body>
</html>
```

将这个文件存为3.aspx，然后通过浏览器透过服务器访问这个文件，将会得到如图2-3的结果。