

第 1 章

ASP.NET 基础

【章节要点】

- .NET Framework类库的命名空间
- Visual Studio 2010、SQL Server 2008软件安装与配置
- HTML基础知识
- JavaScript技术

ASP.NET 基础主要讲解.NET Framework 类库，搭建 ASP.NET 开发环境，HTML 页面布局与控件应用，JavaScript 函数定义与事件引用等。

通过本章的学习，读者将了解.NET 的技术应用，熟知 Visual Studio 2010 和 SQL Server 2008 的软件安装与配置的操作步骤，掌握必要的 HTML 基础知识和 JavaScript 技术。

1.1 .NET 概述

一个.NET 应用是一个使用.NET Framework 类库来编写的，并运行于公共语言运行时（Common Language Runtime，CLR）之上的应用程序。.NET 框架的功能都是通过一个巨大的托管类型集合公开的。这些类型被组织成有等级的命名空间，并且打包到一组程序集中，与 CLR 一起组成了.NET 平台。.NET 类型直接被 CLR 使用，并且是托管宿主环境必不可少的。这些类型处于 mscorlib.dll 程序集中，包括 C#的内置类型以及基本的集合类、流处理类型、序列化、反射、线程和本地互用性。

1.1.1 CLR 与框架核心

1. 系统类型

大多数基本类型都直接在 System 命名空间中，包括 C#的内置类型、Exception 基类、Enum、Array 和 Delegate 基类，以及 Nullable、Type、DateTime、TimeSpan 和 Guid。System 命名空间还包括实现数学运算（Math）、产生随机数字（Random）以及实现各种类型之间的转换（Convert 和 BitConverter）的类型。

2. 文本处理

System.Text 命名空间包括 StringBuilder 类（可编辑的或者可改变的 string），以及用于

文本编码的类型，例如 UTF-8（Encoding 及其子类）。

`System.Text.RegularExpressions` 命名空间包括实现高级的基于模式的查询和替换操作的类型。

3. 集合

.NET 框架提供了多种管理集合元素的类，包括基于链表和基于字典两种结构，并且与统一二者共有特性的一组标准接口协同工作。所有的集合类型都定义在如下所示的命名空间中：

```
System.Collections
System.Collections.Generic
System.Collections.Specialized
System.Collections.ObjectModel
```

4. 查询

语言集成查询（Language Integrated Query，LINQ），允许在本地和远程集合上执行类型安全查询。LINQ 的一个巨大优点是它在多种数据存储之间提供了一致的查询 API。LINQ to SQL API 补充了 `System.Data` 命名空间中的低层的 ADO.NET 类型。解析 LINQ 查询的类型位于如下命名空间中：

```
System.Linq
System.Xml.Linq
System.Data.Linq
System.Linq.Expressions
```

5. XML

XML 被广泛用于 .NET 框架之中，同时也被框架支持，它是一个轻量级 XML 文档对象模型，可以被构造和通过 LINQ 查询。XML 命名空间有：

```
System.Xml
System.Xml.Linq
System.Xml.Schema
System.Xml.XPath
System.Xml.Xsl
System.Xml.Serialization
```

6. 流与输入/输出

.NET 框架为底层输入/输出提供了基于流模型。流（Streams）典型用于直接读/写文件和网络连接，并且可以被链接或者包装到修饰流中以添加压缩或者加密功能。输入/输出类型定义在如下命名空间中：

```
System.IO
System.IO.Pipes
System.IO.Compression
System.IO.IsolatedStorage
```

7. 网络

使用 `System.Net` 命名空间中的类型可以直接访问标准网络协议，例如 HTTP、FTP、TCP/IP 和 SMTP。涉及的命名空间有：

```
System.Net
System.Net.Mail
System.Net.Sockets
```

8. 序列化

.NET 框架提供了几个方法用于将对象保存为二进制或者文本表达形式，同时也提供了将对象从这种表达形式中恢复的方法。这些方法是分布式应用技术所必需的，例如 WCF（Windows Communication Foundation）、Web 服务和远程处理（Remoting），同时也是将对象保存到文件、从文件恢复对象操作所必需的。序列化类型存在于下列命名空间中：

```
System.Runtime.Serialization
System.Runtime.Serialization.Formatters.Binary
System.Runtime.Serialization.Formatters.SOAP
System.Xml.Serialization
```

9. 程序集、反射和属性

C# 程序编译到一个程序集中，程序集包含可执行指令（存储为中间语言 IL 的形式）和描述程序的类型、成员和属性的元数据。使用反射可以在运行时查看元数据，也可以通过动态调用方法实现。使用 `Reflection.Emit` 可以随时创建新的代码。用于反射和处理程序集的类型存在于如下命名空间中：

```
System.Reflection
System.Reflection.Emit
```

10. 安全

.NET 框架提供了自身的安全层，允许将其他程序集封装到沙箱中，也允许将自身封装到沙箱中。涉及的类定义在如下命名空间中：

```
System.Security
System.Security.Permissions
System.Security.Policy
System.Security.Cryptography
```

11. 线程和异步方法

多线程允许平行地执行代码。所有用于线程的特性都在 `System.Threading` 命名空间中。

12. 应用程序域

CLR 在进程中提供了一个附加的隔离层，被称做应用程序域。`AppDomain` 类型被定义在 `System` 命名空间中。

13. 本地互操作性

本地互操作性可以通过平台调用 (P/Invoke) 系统与本地的 Win32 代码相互操作。 .NET 运行时允许调用本地函数、寄存器回调函数、映射数据结构以及与本地数据类型相互操作。支持这种功能的类型在 System.Runtime.InteropServices 命名空间中。

14. 诊断

描述如何与其他进程交互、如何编写 Windows 事件日志以及如何使用用于监测的性能计数器。这些类型被定义在 System.Diagnostics 命名空间中。

1.1.2 .NET 应用技术

1. 用户接口技术

(1) ASP.NET

用 ASP.NET 编写的应用程序借助于 Windows IIS，可以从几乎所有的网络浏览器访问这些应用程序。其优势如下：

- ☐ 不需要部署客户终端。
- ☐ 客户端可以运行于非Windows平台。
- ☐ 非常容易配置更新。

更进一步说，因为在 ASP.NET 应用程序中所编写的大多数代码运行在服务器上，所以可以设计运行在相同应用程序域中的数据访问层，而不需要限制安全性或者可扩展性。相反，实现相同功能的客户端通常没有客户端的安全性或者可扩展性强。对于客户端的解决方法是在客户端和数据库之间插入一个中间层。中间层运行在一个远程应用服务器上（经常与数据库服务器在一起），通过 WCF、Web 服务或者远程处理（Remoting）与客户端通信。

可以通过客户端脚本或者类似 AJAX 技术来提高交互性和响应速度。编写 ASP.NET 应用程序的类型在 System.Web.UI 命名空间及其子命名空间中，并且被打包至 System.Web.dll 程序集。

(2) Windows Presentation Foundation (WPF)

WPF 是一种 UI 框架，可创建丰富的交互式客户端应用程序。WPF 开发平台可支持一组广泛的应用程序开发功能，包括应用程序模型、资源、控件、图形、布局、数据绑定、文档以及安全性。

WPF 的优点如下：

- ☐ 支持丰富的绘图功能，例如任意变换、3D渲染和真正的透明。
- ☐ 其主要度量单位不是基于像素的，所以应用程序可以正确显示任意DPI（每英寸点数）。
- ☐ 有可扩展的动态布局支持，也就是可以局部化应用程序，而不会出现元素重叠的危险。

- ❑ 渲染使用DirectX并且速度很快，很好地利用了图形硬件加速功能。
- ❑ 用户接口可以被公开地描述在XAML文件中，可以被“代码分离”文件独立维持，这样就帮助实现了外观与功能的分离。

用于编写 WPF 应用程序的类型在 System.Windows 命名空间及其子命名空间中。

(3) Windows Forms

Windows Forms 是微软.NET 开发框架的图形用户界面的一部分，该组件通过将现有的 Windows API (Win32 API) 封装为托管代码，提供了对 Windows 本地 (native) 组件的访问方式。

Windows Forms 类型在 System.Windows.Forms (在 System.Windows.Forms.dll 中) 和 System.Drawing (在 System.Drawing.dll 中) 命名空间中。

2. 后台技术

(1) ADO.NET

ADO.NET 是托管数据访问 API。尽管名字是从 ADO (ActiveX Data Objects) 继承而来，但是技术是完全不一样的。ADO.NET 有以下两个主要组件。

- ❑ 数据提供层：数据提供模型为低层访问数据库提供者定义通用类和接口。这些接口包括连接、命令、适配器和读取器 (数据库上的前向、只读指针)。框架中有对 Microsoft SQL Server 和 Oracle 的本地支持，并且拥有 OLE-DB 和 ODBC 提供者。
- ❑ DataSet 模型：DataSet 是一个结构化的数据缓存，类似原始的内存驻留数据库，定义了 SQL 结构，例如表、行、列、关系、约束以及视图。对数据缓存编程，可以减少访问服务器的次数，增加客户端用户接口服务器的可扩展性和响应速率。

DataSet 可以序列化并且被设计为在客户端和服务端应用程序之间传送。

(2) Windows Workflow

Windows Workflow 是一个用于模型化和管理潜在的长期业务进程的框架。Workflow 目标是标准运行时库，提供一致性和互操作性。

(3) COM+ 与 MSMQ

框架允许与 COM+ 相互操作以实现如分布式事务处理这样的服务，使用 System.EnterpriseServices 命名空间中的类型实现。框架还支持 MSMQ (Microsoft Message Queuing, 消息队列) 以实现异步、单向信息传送，使用 System.Messaging 命名空间中的类型实现。

3. 分布式系统技术

Windows Communication Foundation (WCF) 是 .NET 的通信基础结构，有足够的灵活性和可配置性，使得其前一代技术 (远程处理 (Remoting) 和 (.ASMX) Web 服务) 多数情况下是多余的。

WCF、远程处理 (Remoting) 和 Web 服务有些相似，都实现了允许客户端和服务端应用程序通信的基础模型。在服务器端，需要指出远程客户端应用程序可以调用的方法，在客户端，指定或者推断所要调用的服务器方法的签名。在服务器和客户端都需要选择传输

和通信协议（在 WCF 中，这是通过绑定实现的）。客户端建立与服务器的连接，调用远程方法，方法在服务器上运行。

WCF 通过服务协定和数据协定进一步分离客户端与服务器。在概念上，客户端发送一个 XML 消息给远程服务的一个端点，而不是直接调用一个远程方法。这种分离的好处是客户端没有对 .NET 平台或者专有通信协议的依赖。

为 WCF 通信使用的类型在 `System.ServiceModel` 以及它以下的命名空间中。

1.1.3 .NET Framework 4.0 新功能

1. ASP.NET 中的改进

- (1) 改进 JavaScript 用户界面模板和 AJAX 中的数据绑定功能。
- (2) 改进 ASP.NET 的图表控制。

2. WPF 中的改进

- (1) 在 WPF 中增加 Windows 7 多触控、ribbon 控制、任务栏扩展功能支持。
- (2) 在 WPF 中增加 Surface 2.0 SDK 支持。
- (3) 新的业务线控制，包括图表控制、智能编辑、数据网格，以及其他能够提高“建立数据中心的程序”的开发者经验的控制。
- (4) 提高性能和可扩展性。
- (5) 在文字、像素布局、本地化和互操作性中添加可视化改进。

3. Windows Workflow (WF) 中的改进

- (1) 改善活动规划模型。
- (2) 改进设计体验。
- (3) 一个新的流程建模方式。
- (4) 扩大调色板的活动范围。
- (5) 工作流程规则一体化。
- (6) 新邮件相关的功能。

4. WCF 的改进

- (1) 改进 Windows 通信基础 (WCF) 构架，例如，流服务相关支持、双向通信的持久性等。
- (2) 提供了新的 WCF 功能，例如，服务发现、路由器服务、简化配置、REST 支持、诊断和性能提升等。

1.2 运行环境要求和软件安装

1.2.1 硬件和操作系统要求

硬件要求：CPU 推荐使用 1GHz 以上，内存推荐 2GB 以上，显示器推荐 1024×768 像

素，增强色为 16 位。

操作系统要求：应为 Windows Server 2003 Service Pack 2、Windows Server 2008、Windows 7 或 Windows 8 等。

1.2.2 Visual Studio 2010 软件安装

Visual Studio 2010 的安装与 Visual Studio 2008 不同，例如，必须重启后才能继续安装，而且由于安装的项目很多，所以时间也比安装 Visual Studio 2008 长。下面介绍具体的安装流程。

(1) 双击 Visual Studio 2010 的 Setup.exe 文件，系统弹出 Visual Studio 2010 的产品安装选择界面，如图 1.1 所示，选择“安装 Microsoft Visual Studio 2010”选项。

(2) 安装界面以蓝色为基调，如图 1.2 所示，加载安装组件。加载完成后，出现如图 1.3 所示的界面。单击“下一步”按钮，接受安装许可条款，如图 1.4 所示，选中“我已阅读并接受许可条款”单选按钮，单击“下一步”按钮，选择要安装的功能。



图 1.1 安装选择界面



图 1.2 加载安装组件界面



图 1.3 组件加载完成



图 1.4 接受许可条款

(3) 在安装选择界面中，用户可以选择“完全”或“自定义”安装方式，若选择“完全”，安装程序会安装系统的所有功能，如图 1.5 所示。若选择“自定义”，用户可以选择

安装项目，增加了安装程序的灵活性。

(4) 在图 1.5 中，可在选择产品安装路径后（应先考虑路径所在磁盘的剩余空间，再选择安装路径），单击“安装”按钮，进入产品安装界面，如图 1.6 所示。在安装完成 Microsoft .NET Framework 后，安装程序要求“重启”计算机，重启后继续安装。

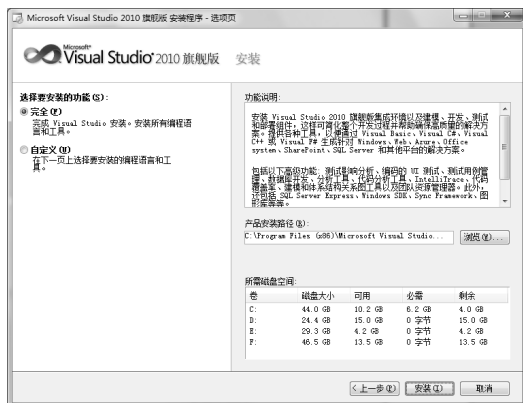


图 1.5 选择安装功能



图 1.6 安装程序界面

(5) 安装完毕后，单击“下一步”按钮，弹出如图 1.7 所示界面。单击“完成”按钮，Visual Studio 2010 程序开发环境安装成功。

(6) 选择“开始”→“所有程序”→Microsoft Visual Studio 2010 命令，弹出如图 1.8 所示对话框，进行首次启动环境设置。



图 1.7 安装完成界面



图 1.8 首次启动环境设置

1.2.3 SQL Server 2008 安装

安装 SQL Server 2008 的流程如下。

(1) 打开 SQL Server 2008 安装文件夹，双击 setup.exe 文件，进入“SQL Server 安装中心”界面，如图 1.9 所示，选择“全新 SQL Server 独立安装或向现有安装添加功能”选项，弹出“安装程序支持规则”界面，如图 1.10 所示。

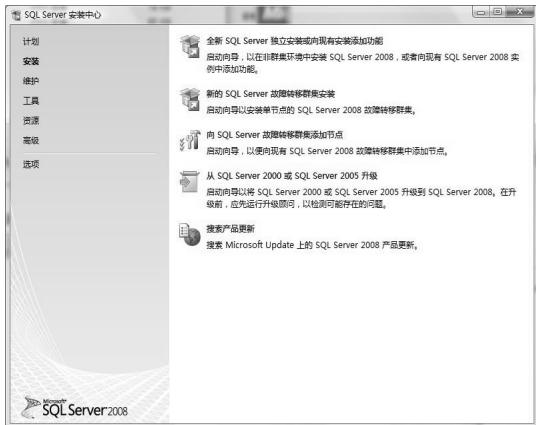


图 1.9 安装中心界面



图 1.10 安装程序支持规则界面

(2) 在安装程序支持规则过程中，不能出现任何失败项，否则安装程序不能正常进行。若全部通过，单击“下一步”按钮，进行功能选择。

(3) 在“功能选择”界面中，对于只安装数据库服务器来说，功能的选择上可以按实际工作需要来制定，一般选择：数据库引擎服务、客户端工具连接、SQL Server 联机丛书、管理工具-基本、管理工具-完整。其中，数据库引擎服务是 SQL 数据库的核心服务，Analysis 及 Reporting 服务可按部署要求安装，这两个服务可能需要 IIS 的支持，如图 1.11 所示。

(4) 在“功能选择”界面，可选择程序安装路径（考虑磁盘存储空间）。默认磁盘为 C，可选择磁盘 D。单击“下一步”按钮，弹出的界面如图 1.12 所示，进行实例配置，可直接选择“默认实例”进行安装，若同一台服务器中有多个数据服务实例，可按不同实例名进行安装。在此，选择“默认实例”，如图 1.12 所示。



图 1.11 功能选择界面

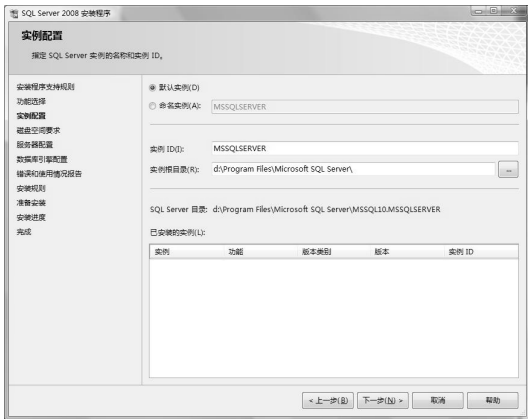


图 1.12 实例配置界面

(5) 单击“下一步”按钮，弹出“服务器配置”界面。服务器配置主要是服务启动账户的配置，服务的账户名推荐使用 NT AUTHORITY\SYSTEM 的系统账户，并指定当前选择服务的启动类型，如图 1.13 所示。

(6) 完成服务器配置后,单击“下一步”按钮,弹出“数据库引擎配置”界面的数据目录选项,如图 1.14 所示,设置完安装目录。选择“账户设置”项,如图 1.15 所示,设置 SQL 登录验证模式及账户密码,身份验证模式推荐使用“混合模式”进行验证,在安装过程中内置的 SQL Server 系统管理员账户(sa)的密码比较特殊,SQL 2008 对 sa 的密码强度要求相对比较高,需要由大小写字母、数字及符号组成,否则将不允许继续安装。在“指定 SQL Server 管理员”栏中建议指定本机的系统管理员 administrator,设置完成后,单击“下一步”按钮,弹出“准备安装”界面,如图 1.16 所示。



图 1.13 服务器配置界面



图 1.14 数据库引擎配置界面数据目录



图 1.15 “数据库引擎配置”界面账户设置



图 1.16 “准备安装”界面

(7) 检查安装项，单击“下一步”按钮，弹出“安装进度”界面，如图 1.17 所示，开始安装。

(8) 安装完成后, 选择“开始”→SQL Server Management Studio 命令, 如图 1.18 所示, 启动 SQL Server。

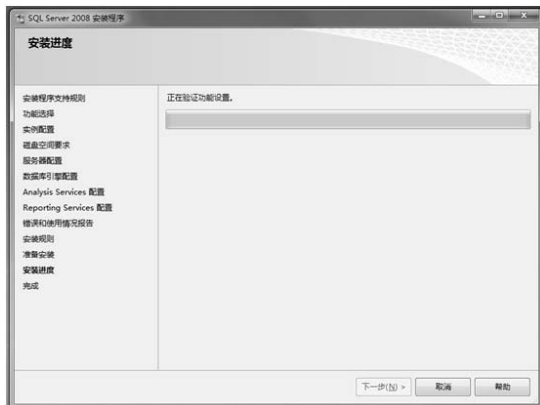


图 1.17 安装进度界面



图 1.18 启动 SQL Server

1.3 Web 编程基础

1.3.1 HTML 技术

HTML (HyperText Markup Language, 超文本标记语言), 用其编写的文件的扩展名是.html 或.htm, 是可供浏览器解释浏览的文件格式。可以使用记事本、写字板或 FrontPage Editor 等编辑工具编写 HTML 文件。HTML 语言使用标签对的方法编写文件, 既简单又方便, 通常使用<标签名></标签名>表示标签的开始和结束 (例如<html></html>), 因此在 HTML 文档中这样的标签对都必须成对使用。HTML 的基本标签介绍如下。

1. <html></html>

<html>标签用于 HTML 文档的最前面, 用来标识 HTML 文档的开始。而</html>恰恰相反, 放在 HTML 文档的最后面, 用来标识 HTML 文档的结束, 两个标签必须一同使用。

2. <head></head>

<head>和</head>构成 HTML 文档的开头部分, 在此标签对之间可以使用<title></title>、<script></script>等标签对, 这些标签对都是描述 HTML 文档相关信息的标签对, <head></head>标签对之间的内容不会在浏览器的框内显示。

3. <body></body>

<body></body>是 HTML 文档的主体部分，在此标签对之间可包含<p>、</p>、<h1>、</h1>、
、<hr>等标签，它们所定义的文本、图像等将会在浏览器的框内显示出来。<body>标签中还包含以下属性，如表 1.1 所示。

表 1.1 <body>标签中常用属性

属 性	用 途	示 例
<body bgcolor="#rrggbb">	设置背景颜色	<body bgcolor="red">红色背景
<body text="#rrggbb">	设置文本颜色	<body text="#0000ff">蓝色文本
<body link="#rrggbb">	设置超链接颜色	<body link="blue">超链接为蓝色
<body vlink="#rrggbb">	设置已使用超链接的颜色	<body vlink="#ff0000">
<body alink="#rrggbb">	设置被单击的超链接的颜色	<body alink="yellow">

说明：以上各个属性可以结合使用，如<body bgcolor="red" text="#0000ff">。引号内的 rrggbb 是用 6 个十六进制数表示的 RGB（即红、绿、蓝三色的组合）颜色，如#ff0000 对应的是红色。此外，还可以使用 HTML 语言所给定的常量名来表示颜色：Black、White、Green、Maroon、Olive、Navy、Purple、Gray、Yellow、Lime、Agua、Fuchsia、Silver、Red、Blue 和 Teal，例如<body text="Blue">表示<body></body>标签对中的文本使用蓝色显示在浏览器的框内。

4. <title></title>

使用过浏览器的用户可能都会注意到浏览器窗口最上边蓝色部分显示的文本信息，那些信息一般是网页的“主题”，若要将所制作网页的主题显示到浏览器的顶部方法很简单，只要在<title></title>标签对之间加入要显示的文本即可。注意，<title></title>标签对只能放在<head></head>标签对之间。

5. <p></p>

<p></p>是用来创建一个段落，在此标签对之间加入的文本将按照段落的格式显示在浏览器上。另外，<p>标签还可以使用 align 属性，用来说明对齐方式，语法是：<p align=""></p>。align 可以是 Left（左对齐）、Center（居中）和 Right（右对齐）3 个值中的任何一个。如<p align="Center"></p>表示标签对中的文本使用居中的对齐方式。

6.

是一个很简单的标签，没有结束标签，因为它用来创建一个回车换行。在
的使用上还有一定的技巧，如果把
加在<p></p>标签对的外边，将创建一个大的回车换行，即
前边和后边的文本的行与行之间的距离比较大，若放在<p></p>的里边则
前边和后边的文本的行与行之间的距离将比较小。

7. <blockquote></blockquote>

在<blockquote></blockquote>之间加入的文本将会在浏览器中按两边缩进的方式显示。

8. <dl></dl>、<dt></dt>和<dd></dd>

<dl></dl>用来创建一个普通的列表,<dt></dt>用来创建列表中的上层项目,<dd></dd>用来创建列表中的最下层项目,<dt></dt>和<dd></dd>都必须放在<dl></dl>标签对之间。

9. 、和

用来创建一个标有数字的列表,用来创建一个标有圆点的列表,只能在或之间使用,此标签对用来创建一个列表项,若放在之间,则每个列表项加上一个数字,若在之间,则每个列表项加上一个圆点。

10. <div></div>

<div></div>用来排版大块 HTML 段落,也用于格式化表,此标签对的用法与<p></p>非常相似,同样有 align 对齐方式属性。

下面举例说明各个标签在 HTML 文档中的应用。

【例 1.1】新建一个网站,在默认页面中设置背景和文本颜色,运行结果如图 1.19 所示。页面代码如下:

```
<html>
<head>
<title>显示在浏览器最上边红色条中的文本</title>
</head>
<body bgcolor="red" text="blue">
<p>红色背景、蓝色文本</p>
</body>
</html>
```

【例 1.2】新建一个网站,在默认页面中添加一个城市列表,运行结果如图 1.20 所示。页面代码如下:

```
<html>
<head>
<title>城市列表</title>
</head>
<body text="blue">
<dl>
<dt>中国城市</dt>
<dd>北京 </dd>
<dd>上海 </dd>
<dd>广州 </dd>
<dt>美国城市</dt>
<dd>华盛顿 </dd>
<dd>芝加哥 </dd>
<dd>纽约 </dd>
</dl>
</body>
</html>
```

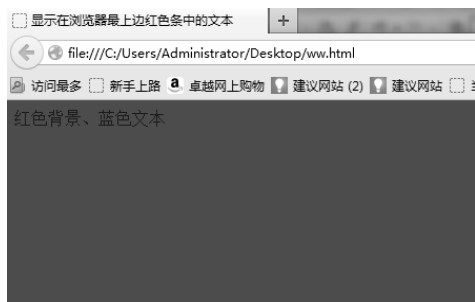


图 1.19 背景和文本颜色

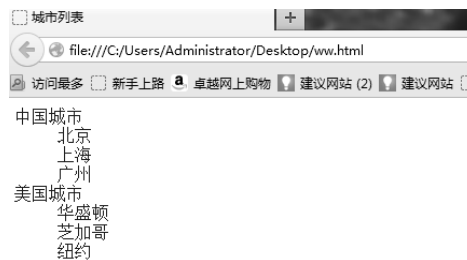


图 1.20 无符号列表

【例 1.3】新建一个网站，在默认页面中创建标有数字或圆点项目符号的列表，在浏览器中显示效果如图 1.21 所示。

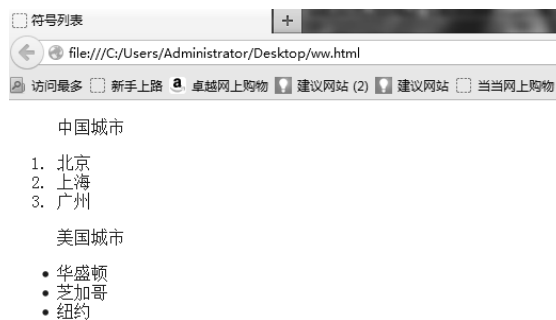


图 1.21 符号列表显示效果

页面代码如下：

```
<html>
<head>
<title>符号列表</title>
</head>
<body text="blue">
<ol>
<p>中国城市 </p>
<li>北京 </li>
<li>上海 </li>
<li>广州 </li>
</ol>
<ul>
<p>美国城市 </p>
<li>华盛顿 </li>
<li>芝加哥 </li>
<li>纽约 </li>
</ul>
</body>
</html>
```

11. 常用 HTML 控件说明（见表 1.2）

表 1.2 常用 HTML 控件说明

控 件 类 型	描 述
<input type="text" />	文本输入框
<input type="submit" />	提交按钮
<input type="reset" />	重置按钮
<input type="radio" />	单选按钮
<input type="password" />	密码输入框
<input type="image" />	图像
<input type="hidden" />	隐藏域
<input type="file" />	文件提交
<input type="checkbox" />	复选框
<input type="button" />	普通按钮
<select><option>选择列表</option></select>	下拉子菜单
<textarea>文本区域</textarea>	文本标签
<label>标签</label>	静态标签

【例 1.4】应用 HTML 控件设计学生信息登记页面，如图 1.22 所示。

学号: 姓名:

性别: ☒ 男 ☐ 女

班级:

备注:

图 1.22 学生信息登记页面

页面代码如下：

```
<body>
<label>学号: </label>
<input type="text" runat="server" width="80px" id="stu_id" />
<label>姓名: </label>
<input type="text" runat="server" width="80px" id="stu_name" /><br>
<label>性别: </label>
<labe>
<input type="radio" runat="server" name="sex" checked="true" value="男" id="stu_sex1"/>男
</label>
<labe><input type="radio" runat="server" name="sex" value="女" id="stu_sex2" />女</label><br>
<labe>班级</labe>
<select><option>13 级计算机科学与技术 1 班</option>
<option>13 级电子信息工程 1 班</option></select><br>
<labe>备注:</labe><br>
<textarea>sdfsf</textarea>
</body>
```

1.3.2 JavaScript 技术

JavaScript 是由 Netscape 公司创造的一种脚本语言。为便于推广，被定为 JavaScript，但是 JavaScript 与 Java 是两门不相干的语言，作用也不一样。作为一门独立的编程语言，JavaScript 在网络上应用广泛，几乎所有的动态网页里都能找到它的身影。但它最主流的应用还是在 Web 上创建动态网页（网页特效）。目前流行的 AJAX 也是依赖于 JavaScript 而存在的。

概括地说，JavaScript 就是一种基于对象和事件驱动，并具有安全性能的脚本语言。脚本语言，简单理解就是在客户端的浏览器上就可以互动响应处理程序的语言，而不需要服务器的处理和响应。当然 JavaScript 也可以做到与服务器的交互响应，而且功能也很强大。而相对的服务器语言例如 ASP、ASP.NET、PHP、JSP 等需要将命令上传到服务器，由服务器处理后回传处理结果。对象和事件是 JavaScript 的两个核心。

JavaScript 可以被嵌入到 HTML 文件中，不需要经过 Web 服务器就可以对用户操作作出响应，使网页的互动性更强；在利用客户端个人计算机资源的同时，适当减小了服务器端的压力，并减少了用户等待时间。

【例 1.5】 创建一网页，应用 JavaScript 控制输出，如图 1.23 所示。

页面代码如下：

```
<html>
<head>
<title>简单的 JavaScript 应用</title>
<script type="text/javascript">
    document.write("JavaScript 第一个实例!");           //在浏览器内直接显示
    alert("继续努力，加油!");                          //弹窗显示
    //在控制台（console）里显示，需要先开启开发工具控制台
    console.log("刻苦学习，勤奋踏实!");
</script>
</head>
<body>
    //网页内容
</body>
</html>
```

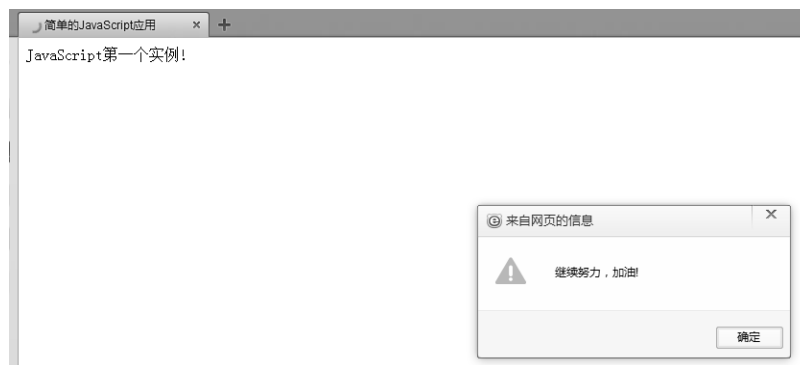


图 1.23 JavaScript 输出界面

1. JavaScript 的使用

HTML 的<script>标签用于把 JavaScript 插入 HTML 页面当中。页面中的脚本必须位于<script>与</script>标签之间。<script>标签中也可使用 type="text/javascript"属性，脚本可放置在 HTML 页面的<head>、<body>标签中。

在 HTML<body>标签中的 JavaScript 会在页面加载的时候被执行。放在<body>标签部分的脚本通常被用来生成页面的内容。在 HTML<head>标签中的 JavaScript 在被调用的时候才执行。当把脚本放在<head>标签中时，可以保证脚本在任何调用之前被加载。

2. JavaScript 函数定义与常用页面事件

JavaScript 函数是由事件驱动的或者当它被调用时执行的可重复使用的代码块。

(1) JavaScript 函数定义

函数定义使用关键字 function，函数名区别大小写，函数名用户自定义符合标识符定义，格式如下：

```
function 函数名(参数表列)
{
    //定义变量
    var returnvalue;
    //这里是要执行的代码
    returnvalue= argument1+ argument2;
    //返回值
    return returnvalue;
}
```

JavaScript 函数调用格式如下：

```
var result = 函数名(参数表列);
```

在 JavaScript 函数内部声明的变量（使用 var）是局部变量，所以只能在函数内部访问它。在不同的函数中可使用名称相同的局部变量，因为只有声明过该变量的函数才能识别出该变量。

在函数外声明的变量是全局变量，网页上的所有脚本和函数都能访问它。

(2) 常用页面事件

常用页面事件如表 1.3 所示。可以在事件发生时执行 JavaScript 函数。

表 1.3 常用页面事件

事 件	事件发生在何时
onabort	图像的加载被中断
onblur	元素失去焦点
onchange	域的内容被改变
onclick	当用户单击某个对象时调用的事件句柄
ondblclick	当用户双击某个对象时调用的事件句柄
onerror	在加载文档或图像时发生错误

续表

事 件	事件发生在何时
onfocus	元素获得焦点
onkeydown	某个键盘按键被按下
onkeypress	某个键盘按键被按下并松开
onkeyup	某个键盘按键被松开
onload	一张页面或一幅图像完成加载
onmousedown	鼠标按钮被按下
onmousemove	鼠标被移动
onmouseout	鼠标从某元素移开
onmouseover	鼠标移到某元素之上
onmouseup	鼠标按键被松开
onreset	重置按钮被单击
onresize	窗口或框架被重新调整大小
onselect	文本被选中
onsubmit	确认按钮被单击
onunload	用户退出页面

**注意**

onload 和 onunload 事件会在用户进入或离开页面时被触发。onchange 事件常结合对输入字段的验证来使用。onmouseover 和 onmouseout 事件可用于在用户的鼠标移至页面元素上方或移出元素时触发函数。Onmousedown、onmouseup 以及 onclick 构成了鼠标单击事件的所有部分。首先当单击鼠标按钮时, 会触发 onmousedown 事件, 当释放鼠标按钮时, 会触发 onmouseup 事件, 最后, 当完成单击时, 会触发 onclick 事件。

3. JavaScript 消息框函数的引用

可以在 JavaScript 中创建 3 种消息框: 警告框、确认框、提示框。

(1) alert("文本")

警告框经常用于确保用户可以得到某些信息。当警告框出现后, 用户需要单击确定按钮才能继续进行操作。如图 1.24 所示, 页面代码如下:

```
<html>
<head>
<script type="text/javascript">
function Disp_Alert()
{
    alert("湖水深, 不能在此戏水, 危险!")
}
</script>
</head>
```

```

<body>
<input type="button" onclick="Disp_Alert()" value="显示警告窗口" />
</body>
</html>

```

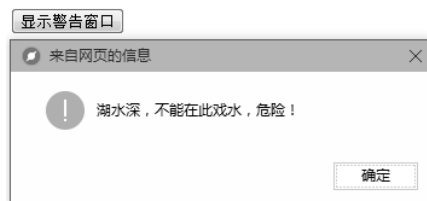


图 1.24 显示警告窗口

(2) confirm("文本")

确认框用于使用户可以验证或者接受某些信息。当确认框出现后，用户需要单击“确定”或者“取消”按钮才能继续进行操作。如果用户单击“确定”按钮，那么返回值为 `true`；如果用户单击“取消”按钮，那么返回值为 `false`。显示确认窗口如图 1.25 所示，单击“确定”按钮后弹出的对话框如图 1.26 所示，页面代码如下：

```

<html>
<head>
<script type="text/javascript">
function Show_Confirm()
{
var r=confirm("请单击按钮！");
if (r==true)
{
alert("已确认！");
}
else
{
alert("已取消！");
}
}
</script>
</head>
<body>
<input type="button" onclick="Show_Confirm()" value="显示确认窗口" />
</body>
</html>

```

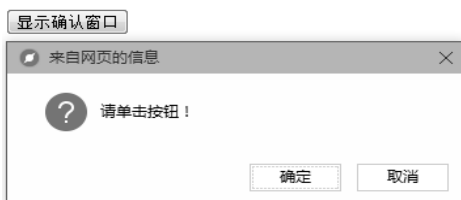


图 1.25 显示确认窗口

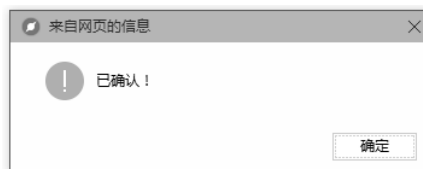


图 1.26 单击“确定”按钮后弹出对话框

(3) prompt("文本","默认值")

提示框经常用于提示用户在进入页面输入某个值。当提示框出现后，用户需要输入某个值，然后单击“确认”或“取消”按钮才能继续操作。如果用户单击“确认”按钮，那么返回值为输入的值。如果用户单击“取消”按钮，那么返回值为 null。如图 1.27 所示，页面代码如下：

```
<html>
<head>
<script type="text/javascript">
function Disp_Prompt()
{
    var nationname=prompt("请输入您的国籍","中国")
    if (nationname!=null && nationname!="")
    {
        document.write("你的国籍是" + nationname);
    }
}
</script>
</head>
<body>
<input type="button" onclick="Disp_Prompt()" value="显示提示框" />
</body>
</html>
```



图 1.27 显示提示窗口

4. JavaScript 计时函数引用

通过调用 setTimeout()函数完成 JavaScript 计时事件，通过调用 clearTimeout()函数取消计时事件。

(1) 单击“确定”按钮后，会在 3 秒后弹出警告框，如图 1.28 所示，页面代码如下：

```
<html>
<head>
<script type="text/javascript">
function TimedMsg()
{
    var t=setTimeout("alert('3 秒! ')",3000)
}
</script>
</head>
<body>
```

```

<form>
<input type="button" value="显示定时警告框" onclick = "TimedMsg()">
</form>
</body>
</html>

```

(2) 单击“显示计时的文本”按钮，在文本框中连续显示事件内容，如图 1.29 所示，页面代码如下：

```

<html>
<head>
<script type="text/javascript">
function timedText()
{
var t1=setTimeout("document.getElementById('txt_time').value='2 秒'",2000)
var t2=setTimeout("document.getElementById('txt_time').value='5 秒'",5000)
var t3=setTimeout("document.getElementById('txt_time').value='8 秒'",8000)
}
</script>
</head>
<body>
<form>
<input type="button" value="显示计时的文本" onclick="timedText()">
<input type="text" id="txt_time">
</form>
</body>
</html>

```

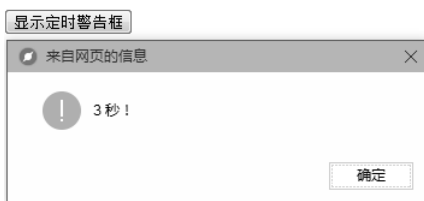


图 1.28 显示定时警告框

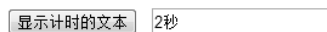


图 1.29 显示计时文本

(3) 一个 JavaScript 小时钟，在加载页面时显示时钟，如图 1.30 所示，页面代码如下：

```

<html>
<head>
<script type="text/javascript">
function StartTime()
{
var today=new Date()
var h=today.getHours()
var m=today.getMinutes()
var s=today.getSeconds()
//add a zero in front of numbers<10
m=checkTime(m)
s=checkTime(s)

```

```

document.getElementById('txt').innerHTML=h+":"+m+": "+s
t=setTimeout('startTime()',500)
}

function CheckTime(i)
{
if (i<10)
    {i="0" + i}
    return i
}
</script>
</head>
<body onload="StartTime()">
<div id="txt"></div>
</body>
</html>

```

13:13:21

图 1.30 显示时钟

5. 房贷计算器设计

大家贷款买房的时候，都希望能借助一款简易方便的计算器，快速地计算出每月的还款额，如图 1.31 所示为房贷计算器。其实这些计算器都是利用 JavaScript 程序设计出来的，以商业贷款计算器为例，讲述利用 JavaScript 实现该计算器。



等额本息还款法

贷款类别: 商业贷款

☐ 根据面积、单价计算
☒ 根据贷款总额计算

计算方式: 贷款总额: 万元

按揭年数: 20年(240期)

利率: 12年07月06日利率下限(85折)

开始计算 重新计算

计算结果:

贷款总额: 元	贷款总额: 元
还款总额: 元	支付利息款: 元
首期付款: 元	贷款月数:
月均还款: 元	

图 1.31 房贷计算器



说明

房贷计算器中，利用了 JavaScript 的函数，如取 n 次方的函数 `Math.pow`，保留小数位的 `Math.Round()` 函数，有兴趣的读者，可以将本计算器的功能扩展成与新浪乐居网的房贷计算器同样的功能。当然，同样道理，利用 JavaScript 也可以实现车贷计算的功能。

设计步骤如下：

(1) 界面设计。为了方便读者理解，此处将图 1.31 中的商业贷款计算器界面设计为如图 1.32 所示的样式。

简易房贷计算器			
贷款类别	商业贷款		
贷款总额			
按揭年数	5年		
利率			
开始计算 重新计算			
计算结果：			
贷款总额：		还款总额：	
支付利息：		月均还款：	

图 1.32 房贷计算器界面设计

界面代码如下：

```
<body>
<div align="center"><strong>简易房贷计算器</strong></div>
<form name="form1">
<table width="542" border="1" align="center">
<tr>
<td width="85"><strong>贷款类别</strong></td>
<td colspan="3"><label for="kind"></label>
<select name="kind" id="kind">
<option value="1">商业贷款</option>
<option value="2">公积金贷款</option>
</select></td>
</tr>
<tr>
<td><strong>贷款总额</strong></td>
<td colspan="3"><label for="zongge"></label>
<input type="text" name="zongge" id="zongge" /></td>
</tr>
<tr>
<td><strong>按揭年数</strong></td>
<td colspan="3"><label for="year"></label>
<select name="year" id="year">
<option value="60">5 年</option>
<option value="120">10 年</option>
<option value="180">15 年</option>
<option value="240">20 年</option>
</select></td>
</tr>
<tr>
<td><strong>利率</strong></td>
<td colspan="3"><label for="rate"></label>
<input type="text" name="rate" id="rate" /></td>
</tr>
<tr align="center">
<td colspan="4">
<input type="button" name="start" id="start" value="开始计算"

```

```

onclick="jisuan()" />
<input type="reset" name="clear" id="clear" value="重新计算" /></td>
</tr>
<tr>
<td colspan="4"><strong>计算结果: </strong></td>
</tr>
<tr>
<td><strong>贷款总额: </strong></td>
<td width="168"><label for="showdaikuan"></label>
<input type="text" name="showdaikuan" id="showdaikuan" /></td>
<td width="91"><strong>还款总额</strong>: </td>
<td width="170"><label for="huankuanzongge"></label>
<input type="text" name="huankuanzongge" id="huankuanzongge" /></td>
</tr>
<tr>
<td><strong>支付利息: </strong></td>
<td><label for="zhifulixi"></label>
<input type="text" name="zhifulixi" id="zhifulixi" /></td>
<td><strong>月均还款: </strong></td>
<td><label for="yuejunhuankuan"></label>
<input type="text" name="yuejunhuankuan" id="yuejunhuankuan" /></td>
</tr>
<tr>
<td colspan="4">&nbsp;</td>
</tr>
</table></form></body>

```

(2) 编写 JavaScript 脚本代码。



注意

月均还款公式：月还款额=本金×月利率×(1+月利率)ⁿ/[(1+月利率)ⁿ-1]。式中，月利率=年利率/12，n 表示贷款月数，ⁿ 表示 n 次方，如⁶⁰，表示 60 次方（贷款 5 年、60 个月）。计算(1+月利率)ⁿ-1 时，要先将小括号内的算出来，乘方后再减 1。

```

<script language="javascript">
function jisuan()
{ var daizongge=document.form1.zongge.value;           //贷款总额
  var daimonth=document.form1.year.value;             //贷款月数
  var rate =document.form1.rate.value;                 //贷款利率
  document.form1.showdaikuan.value=daizongge;          //显示贷款总额
  var rate1=1+rate/12/100;
  var rate2=rate/12/100;
  var temp1=Math.pow(rate1,daimonth)-1;
  var monthhuankuan=daizongge*rate2*Math.pow(rate1,daimonth)/(temp1); //月还款
  monthhuankuan=Math.round(monthhuankuan*100)/100;      //月还款保留小数位后两位
  document.form1.yuejunhuankuan.value=monthhuankuan;
  var huankuanetemp=monthhuankuan*daimonth+daizongge;
  var huankuane= Math.round(huankuanetemp*100)/100;     //还款总额保留到 2 位小数

```

```
document.form1.huankuanzonghe.value=huankuane;
var lixi=Math.round((huankuane -daizonghe)*100)/100; //应还利息也保留 2 位小数
document.form1.zhifulixi.value= lixi; //显示利息
}
</script>
```

(3) 运行调试，结果如图 1.33 所示。

简易房贷计算器

贷款类别	商业贷款		
贷款总额	200000		
按揭年数	15年		
利率	5.56		
开始计算 重新计算			
计算结果:			
贷款总额:	200000	还款总额:	295297.22
支付利息:	95297.22	月均还款:	1640.54

图 1.33 房贷计算器运行结果

1.4 综合案例

在短信平台管理系统中，操作人员发送短信时要输入手机号码，现要求输入手机号码时系统能够智能识别并进行提醒。

要求：在编辑框中只能输入有效手机号码，号码之间自动用“,”隔开，若输入号码错误，系统弹出消息框进行提醒，编辑框中只能输入数字 0~9，如图 1.34 所示。

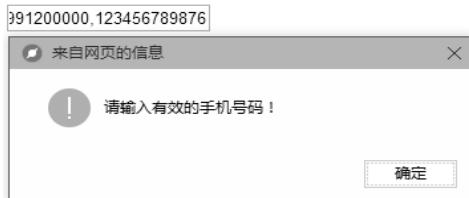


图 1.34 提醒对话框

设计过程：

(1) 在网页中插入一个编辑框，设置属性和事件。代码如下：

```
<html>
<head>
<script type="text/javascript">
</script>
</head>
<body>
<input type="text" onkeydown="javascript:checkNumeric()" onblur="javascript:lasttel_query()"
id="TxtContactPerson"/>
</body>
</html>
```



说明

事件 `onkeydown` 表示某个键盘按键被按下时要执行的相应函数；事件 `onblur` 表示元素失去焦点，也就是离开控件时要执行的函数。

(2) 在 `<head>` 部分添加 `checkNumeric()` 和 `lasttel_query()` 函数，`checkNumeric()` 实现有效字符输入判断、手机号码有效性判断以及手机号码之间添加 “,” 功能，`lasttel_query()` 实现最后一个手机号码有效性判断功能。

`checkNumeric()` 函数代码如下：

```
function checkNumeric() {
    var i = 1;
    var strtel = document.getElementById("TxtContactPerson").value;
    var len = document.getElementById("TxtContactPerson").value.length;
    var myreg = /^(((13[0-9])|(15[0-5])|(18[0-9])|147)+\d{8})$/;
    if (event.keyCode >= 48 && event.keyCode <= 57) {
        if (len % 12 == 11) {
            var strtel1 = strtel.substr(strtel.length - 11, 11);
            if (!myreg.test(strtel1)) {
                alert("请输入有效的手机号码!");
                document.getElementById("TxtContactPerson").value = strtel.substr(0, strtel.length - 11);
                i = 0;
                event.returnValue = false;
            }
            if (i) {
                document.getElementById("TxtContactPerson").value = strtel + ",";
                event.returnValue = true;
            }
        }
    }
    else if (event.keyCode == 46 || event.keyCode == 8)
        event.returnValue = true;
    else
        event.returnValue = false;
}
```

`lasttel_query()` 函数代码如下：

```
function lasttel_query() {
    var strtel = document.getElementById("TxtContactPerson").value;
    var len = document.getElementById("TxtContactPerson").value.length;
    var myreg = /^(((13[0-9])|(15[0-5])|(18[0-9])|147)+\d{8})$/;
    var lastlen = len % 12;
    var strlast = strtel.substr(len - lastlen, lastlen);
    if (lastlen != 11 || !myreg.test(strlast)) {
        alert("最后手机号码有误!");
        document.getElementById("TxtContactPerson").value = strtel.substr(0, len - lastlen - 1);
        document.getElementById("TxtContactPerson").focus();
    }
}
```

(3) 页面全部代码如下:

```
<html>
<head>
<script type="text/javascript">
    function checkNumeric() {
        var i = 1;
        var strtel = document.getElementById("TxtContactPerson").value;
        var len = document.getElementById("TxtContactPerson").value.length;
        var myreg = /^(((13[0-9])|(15[0-5])|(18[0-9])|147)+\d{8})$/;
        if (event.keyCode >= 48 && event.keyCode <= 57) {
            if (len % 12 == 11) {
                var strtel1 = strtel.substr(strtel.length - 11, 11);
                if (!myreg.test(strtel1)) {
                    alert('请输入有效的手机号码! ');
                    document.getElementById("TxtContactPerson").value = strtel.substr(0,
strtel.length - 11);

                    i = 0;
                    event.returnValue = false;
                }
                if (i) {
                    document.getElementById("TxtContactPerson").value = strtel + ",";
                    event.returnValue = true;
                }
            }
        }
        else if (event.keyCode == 46 || event.keyCode == 8)
            event.returnValue = true;
        else
            event.returnValue = false;
    }
    function lasttel_query() {
        var strtel = document.getElementById("TxtContactPerson").value;
        var len = document.getElementById("TxtContactPerson").value.length;
        var myreg = /^(((13[0-9])|(15[0-5])|(18[0-9])|147)+\d{8})$/;
        var lastlen = len % 12;
        var strlast = strtel.substr(len - lastlen, lastlen);
        if (lastlen != 11 || !myreg.test(strlast)) {
            alert('最后手机号码有误! ');
            document.getElementById("TxtContactPerson").value = strtel.substr(0, len - lastlen - 1);
            document.getElementById("TxtContactPerson").focus();
        }
    }
</script>
</head>
<body>
<input type="text" onkeydown="javascript:checkNumeric()" onblur="javascript:lasttel_query()"
id="TxtContactPerson"/>
</body>
</html>
```

(4) 调试程序，运行结果如图 1.34 所示。

1.5 章 节 训 练

1. 应用 HTML 设计课程信息录入界面，包括课程代码、课程名称、开设专业、学分、考试形式等信息。
2. 应用 JavaScript 编写函数验证文本框是否为空。