高等教育规划教材

C#程序设计教程

第 2 版

崔 淼　贾红军　主编

免费提供电子教案、源代码

下载网址 http://www.cmpedu.com

机械工业出版社
CHINA MACHINE PRESS

高等教育规划教材

C#程序设计教程

第2版

崔 淼 贾红军 主 编

机械工业出版社

本书以 Visual Studio 2015为开发平台，采用“任务驱动”方式，全面细致地介绍了 Visual C#的基础知识、特点和具体应用，突出面向对象程序设计思想。本书主要包括 C#、.NET 和 Visual Studio 简介，C#语法基础，流程控制语句与控件，面向对象的程序设计方法，数组、结构与集合，接口、委托和事件，泛型，异常处理、程序调试和文件操作，数据库和数据绑定，创建数据库应用程序，使用 Microsoft Excel 输出报表，以及使用多线程等方面的内容。

本书可作为高等院校计算机专业 C#课程的教材，同时也可作为广大计算机爱好者和各类 Visual C#程序设计培训班的教学用书。

本书配有电子教案，需要的教师可登录www.cmpedu.com 免费注册，审核通过后下载，或联系编辑索取（QQ：2966938356，电话：010-88379739）。

图书在版编目（CIP）数据

C#程序设计教程 /崔淼，贾红军主编. —2 版. —北京：机械工业出版社，2018.2

高等教育规划教材

ISBN 978-7-111-59051-4

Ⅰ. ①C… Ⅱ. ①崔… ②贾… Ⅲ. ①C 语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字（2018）第 018346 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：和庆娣 责任编辑：和庆娣

责任校对：张艳霞 责任印制：常天培

唐山三艺印务有限公司印刷

2018 年 2 月第 2 版·第 1 次印刷

184mm ×260mm ·18.75印张·456 千字

0001－3000 册

标准书号：ISBN 978-7-111-59051-4

定价：55.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

服务咨询热线：（010）88379833

读者购书热线：（010）88379649

网络服务

机 工 官 网：www.cmpbook.com

机 工 官 博：weibo.com/cmp1952

教育服务网：www.cmpedu.com

金 书 网：www.golden-book.com

出 版 说 明

当前，我国正处在加快转变经济发展方式、推动产业转型升级的关键时期。为经济转型升级提供高层次人才，是高等院校最重要的历史使命和战略任务之一。高等教育要培养基础性、学术型人才，但更重要的是加大力度培养多规格、多样化的应用型、复合型人才。

为顺应高等教育迅猛发展的趋势，配合高等院校的教学改革，满足高质量高校教材的迫切需求，机械工业出版社邀请了全国多所高等院校的专家、一线教师及教务部门，通过充分的调研和讨论，针对相关课程的特点，总结教学中的实践经验，组织出版了这套“高等教育规划教材”。

本套教材具有以下特点：

1）符合高等院校各专业人才的培养目标及课程体系的设置，注重培养学生的应用能力，加大案例篇幅或实训内容，强调知识、能力与素质的综合训练。

2）针对多数学生的学习特点，采用通俗易懂的方法讲解知识，逻辑性强、层次分明、叙述准确而精炼、图文并茂，使学生可以快速掌握，学以致用。

3）凝结一线骨干教师的课程改革和教学研究成果，融合先进的教学理念，在教学内容和方法上做出创新。

4）为了体现建设“立体化”精品教材的宗旨，本套教材为主干课程配备了电子教案、学习与上机指导、习题解答、源代码或源程序、教学大纲、课程设计和毕业设计指导等资源。

5）注重教材的实用性、通用性，适合各类高等院校、高等职业学校及相关院校的教学，也可作为各类培训班教材和自学用书。

欢迎教育界的专家和老师提出宝贵的意见和建议。衷心感谢广大教育工作者和读者的支持与帮助！

机械工业出版社

前　言

Visual C#是微软公司 Visual Studio 开发平台中推出的完全面向对象的编程语言。利用这种面向对象的、可视化的编程技术，结合事件驱动的模块设计，将使程序设计变得轻松快捷。因此，Visual C#在国内外各个领域中得到了广泛应用。本书以 Visual Studio 2015 为开发平台，结合大量易于理解的实例，面向无编程基础的读者逐步学习 Visual C#程序设计的整个过程。在叙述上以深入浅出的语言并结合直观的图示、演练，使读者能够轻松地理解面向对象编程的基本概念与思想。

本书注重突出面向对象的程序设计思想，不仅在讲述内容上详细介绍了面向对象的相关概念及编程技巧，而且几乎在所有的演练和实训中都使用“任务驱动”的方式，强调使用面向对象的程序设计方法实现程序功能。强调程序功能由类及其属性、方法等实现，窗体中的控件仅组成用户操作界面（UI）的“松耦合”程序设计方式。

本书共分为 12 章，主要包括 C#、.NET 和 Visual Studio 简介，C#语法基础，流程控制语句与控件，面向对象的程序设计方法，接口、委托和事件，泛型，异常处理、程序调试和文件操作，数据库和数据绑定，创建数据库应用程序，使用 Microsoft Excel 输出报表，以及多线程等内容。

本书编者已在课堂上讲授程序设计语言多年，并参加过许多实际应用系统的开发，拥有丰富的教学经验和实践经验。在内容的处理上，以面向对象的程序设计作为主线，以相关的C#控件作为辅助，通过本书的学习，读者不但能学会面向对象程序设计的基本知识、设计思想和方法，读者还能很容易地过渡到其他面向对象程序设计语言的学习与使用上。

本书由崔淼、贾红军主编，其中崔淼编写第 9、10 章，贾红军编写第 1、4 章，徐鹏编写 5、6 章，朱婷婷编写 2、3 章，赵晓华编写第 7 章，孙民瑞编写第 8 章，刘瑞新编写第 12 章，第 11 章及资料的收集整理、课件的制作由李建彬、刘大学、陈周、骆秋容、刘克纯、缪丽丽、刘大莲、彭守旺、庄建新、彭春芳、崔瑛瑛、翟丽娟、韩建敏、庄恒、徐维维、徐云林、马春锋、孙洪玲、田金雨完成。本书由刘瑞新教授策划并统稿。本书在编写过程中得到了许多一线教师的大力支持，提出了许多宝贵意见，使本书更加符合教学规律，在此一并表示感谢。

由于计算机信息技术发展迅速，书中难免存在不足和疏漏之处，恳请广大读者批评指正。

编　者

目　录

第1章　C#、.NET和Visual Studio简介

2000年6月，Microsoft正式发布了一种全新的软件开发平台——Microsoft .NET Framework（.NET框架），旨在提供一种创建和运行下一代应用程序和Web服务的新方式。Visual C#（简称为C#）是专门为.NET Framework开发的程序设计语言，它借鉴了Delphi、C++及Java的语法及主要功能，是一种类型安全的面向对象通用语言，可用于编写任何类型的应用程序。

Visual Studio是Microsoft推出的一套完整的、基于.NET Framework的应用程序开发工具集。本书所讲解的版本为Visual Studio 2015，内置最高.NET Framework版本为4.6.1，默认使用的.NET Framework版本为4.5.2。同时提供对.NET Framework 2.0、.NET Framework 3.0等版本的支持。使用Visual Studio 2015可以方便地进行Windows、ASP.NET Web和移动端等各类应用程序的开发。

本书使用Visual Studio 2015 Enterprise Update 3和Windows 10专业版作为写作背景，但讲述内容除特别声明外也适用于Visual Studio 2008及以上各版本。

1.1　.NET Framework

.NET Framework提供了一些工具和技术，使得开发人员能够以独立于语言和平台的方式创建、调试和运行应用程序或Web服务，它提供了庞大的类库，用来以简单、高效的方式提供对众多常见业务的支持。

1.1.1　公共语言运行时（CLR）

公共语言运行时（Common Language Runtime，CLR）是所有.NET应用程序运行所必需的环境支持，是所有.NET应用程序的编程基础。可以将CLR看作是一个在执行.NET应用程序时，代码的“管理者”。通常将能够被.NET Framework管理的代码称为“托管代码”（Managed Code），反之称为“非托管代码”（Unmanaged Code）。

CLR主要包含有通用类型系统（CTS）、公共语言规范（CLS）、微软中间语言（MSIL）、虚拟执行系统（VES）、内存管理和垃圾回收等模块。

1. 通用类型系统（CTS）和公共语言规范（CLS）

通用类型系统（Common Type System，CTS）可以使所有基于.NET的程序设计语言（如C#、Visual Basic等）共享相同的类型定义，从而让这些使用不同语言编写的应用程序使用一致的方式操作这些类型。

由于CTS要兼容于不同语言编写的应用程序，所以它需要使用独立于任何一种编程语言的方式来定义类型，并且需要兼顾不同编程语言之间的差异。为此，CTS提供了一个所有.NET语言都必须遵守的、最基本的规则集，称为“公共语言规范”（Common Language Specification，CLS）。

2. 微软中间语言（MSIL）和虚拟执行系统（VES）

CTS和CLS解决了独立于编程语言的问题，使程序开发人员可以自由地选择自己熟悉的

编程语言，甚至可以实现在同一个应用程序项目中使用不同的编程语言进行开发。但如果编译器生成的可执行代码不能摆脱对硬件的依赖，则上述优势将消失殆尽。为了解决这一问题，.NET Framework 在编译托管代码时，并未直接生成可执行的目标代码（本机代码），而是将其编译成一种类似于汇编语言的、能兼容不同硬件环境的通用中间语言（Microsoft Intermediate Language，MSIL）表示的代码。

虚拟执行系统（Virtual Execution System，VES）也称为“托管运行环境”，它是 CLR 的一个重要组成部分，负责处理应用程序所需要的低级核心服务。.NET 应用程序启动时，由 VES 负责加载并执行 MSIL 代码，由“虚拟执行系统”提供的“即时编译器”（Just-In-Time，JIT）在程序运行阶段将 MSIL 转换成本计算机代码。

3. 内存管理

内存管理主要是为应用程序运行分配必要的计算机内存空间，定期回收不再使用的内存空间。在许多早期的编程语言中，这一直都是一个很麻烦的问题，稍有不慎就可能导致程序运行错误。在这些编程语言中，开发人员需要考虑如何在恰当的时间，合理地分配或回收计算机内存，以保证整个系统的正常运行。而在.NET 中这项工作交由 VES 自动完成，一般情况下无须开发人员干预。这种机制提高了应用程序的稳定性，开发人员可以将关注点集中在应用程序所需的业务逻辑设计上，提高了开发效率。

1.1.2 .NET Framework 类库

.NET Framework 类库（Framework Class Library，FCL）是系统预设的、经过编译的、用于实现一些常用功能的程序模块集合。程序员在开发过程中只需简单地调用其中的某些类，即可实现希望的操作，而不必关心该操作是如何实现的。.NET Framework 类对用户来说就像一个“黑匣子”，输入原始数据或请求即可得到加工处理后的数据或实现某种操作，而不必关心黑匣子内部发生了什么。

目前，FCL 中包含 4000 多个公有类，是当今最大的类库之一。数量众多、功能强大的类库使.NET Framework 具有了将复杂问题简单化的特征，极大地提高了应用程序的运行稳定性和开发效率。

在 FCL 中使用“命名空间”（Name Space）将数量众多、功能各异的.NET 类划分为不同的层次。命名空间使用点分式语法表示该层次结构（每层之间使用符号“.”进行分隔）。例如，System.Data.SqlClient 表示 SqlClient 隶属于 Data，而 Data 又隶属于 System。为了减少程序代码的输入量，可以在项目的代码窗口最上方使用 using 语句添加对所需命名空间的引用。图 1-1 所示为新建一个 Windows 窗体项目后，由系统自动添加到项目中的命名空间引用代码。灰色显示的部分表示已添加但并未使用的命名空间引用。

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

图 1-1 引用命名空间

1.1.3 C#项目与.NET Framework 的关系

C#项目与.NET Framework 之间的关系如图 1-2 所示。

可以看出，C#程序需要在.NET Framework 上运行，用 C#语言编写的源代码首先要被编译成 MSIL 中间语言。中间语言代码和资源（位图、字符串等）以“程序集”（可执行文件，扩展名通常为 exe 或 dll）的形式保存在磁盘中。

当执行 C#程序时，程序集将被加载到 CLR 中，如果程序集满足安全要求，CLR 会直接

执行即时编译（JIT），将 MSIL 代码转换成本机指令。此外，CLR 还提供与自动垃圾回收、异常处理和资源管理相关的一些服务。

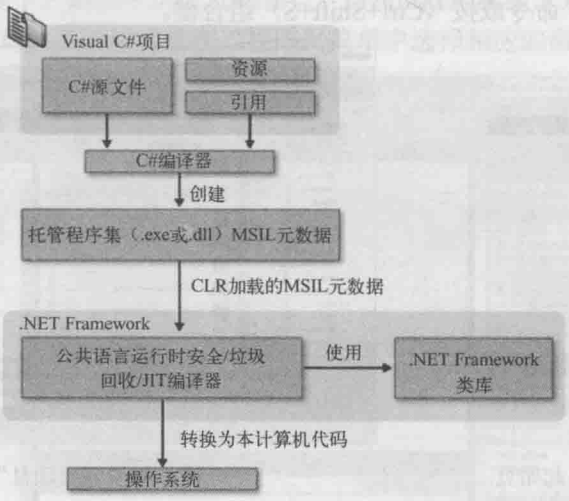

图 1-2　C#项目与.NET Framework 之间的关系

由于 C#编译器生成的中间语言代码符合通用类型系统（CTS），因此由 C#生成的中间语言代码可以与 Visual Basic、Visual C++等符合 CTS 的、20 多种基于.NET Framework 的编程语言生成的代码进行交互。这就使得一个程序集中可能包含多个用不同语言编写的、符合 CTS 规范的模块，并且其中包含的类型也可以相互引用，就像是用同一种语言编写的一样。这一特征被称为“语言互操作性”，是.NET Framework 的一项重要功能。

1.2　Visual Studio 项目管理

Visual Studio 是 Microsoft 专门为.NET Framework 设计的开发平台，它将程序设计中需要的各个环节（程序组织、界面设计、程序设计、运行和调试等）集成在同一个窗口中，极大地方便了开发人员的设计工作。通常将这种集多项功能于一身的开发平台称为“集成开发环境”（Integrated Development Environment，IDE）。

1.2.1　新建和打开项目

在 Visual Studio 中，一个完整的应用系统可能会包含在若干个“项目”中，与应用系统相关的所有项目集合被称为“解决方案”。创建一个 Windows 应用程序，首先需要在 Visual Studio 环境中创建一个新项目。

1. 新建和保存项目

启动 Visual Studio 后首先显示如图 1-3 所示的“起始页”内容，如在“开始”选项组中单击“新建项目”链接，将弹出如图 1-4 所示的“新建项目”对话框，选择 Visual C#模板下的“Windows 窗体应用程序”选项，选择使用的.NET Framework 版本（默认为.NET 4.5.2，最高支持.NET 4.6.1，同时提供对.NET 2.0 和.NET 3.0 的支持），并指定项目名称、保存位置及解决方案名称，然后单击“确定”按钮，系统将根据用户设置自动创建一个“空白”的 Windows 窗体应用程序框架。

创建好一个 Visual C#项目后，用户可以在 IDE 环境中完成界面设计、代码编写和运行调试等。需要保存项目文件时，可单击 Visual Studio 工具栏中的“全部保存”按钮，或选择“文件”→“全部保存”命令或按〈Ctrl+Shift+S〉组合键。

图 1-3　Visual Stutio 起始页

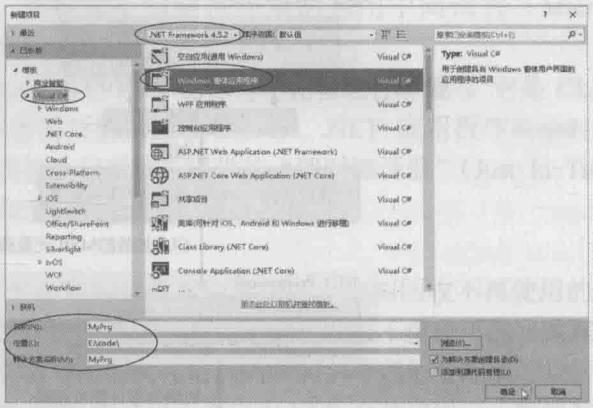

图 1-4　“新建项目”对话框

此外，在关闭 Visual Studio 时系统会检查当前项目中是否有尚未保存的文件，若有，则弹出如图 1-5 所示的对话框，询问用户是否要保存更改。

“项目”（Project，也称为“工程”）是 Visual Studio 用来标识构建应用程序的方式。它作为一个容器对程序的源代码和相关资源文件进行统一管理和组织，项目文件是*.csproj（C#）或*.vbproj（Visual Basic）。

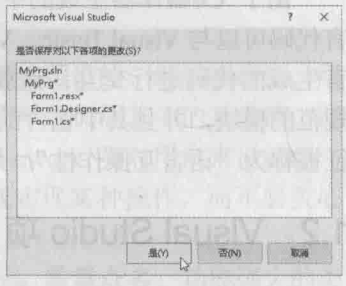

图1-5　保存项目文件

在 Visual Studio 中创建一个项目时，系统会自动创建相应的“解决方案”（Solution），解决方案文件默认与项目同名，其扩展名为 sln。需要注意的是，一个解决方案中可以包含多个项目，可以将解决方案理解为“项目的容器”，将项目理解为“程序的容器”。

2. 打开项目

在 Visual Studio 中创建一个项目时，系统会自动将项目包含到一个解决方案中，并默认为解决方案创建一个文件夹，解决方案文件（*.sln）就存放在该文件夹内。

可以通过以下几种途径打开已创建的项目。

1）在“起始页”的“开始”选项组中单击“打开项目”链接，在弹出的对话框中选择解决方案文件（*.sln）将其打开。

2）通过在 Windows 操作系统的“计算机”或“资源管理器”窗口中直接双击保存在解决方案文件夹中的*.sln 文件将其打开。

3）最近使用过的项目会出现在“起始页”的“最近”选项组中，单击其名称可再次打开到 Visual Studio 集成开发环境中。

1.2.2　集成开发环境中的主要子窗口

创建了一个新项目后，系统进入如图 1-6 所示的 Visual Studio 集成开发环境主界面。从图中可以看到除了具有菜单栏、工具栏外，还包含许多子窗口，其中最常用的是“工具箱”“解

决方案资源管理器”“属性”和“输出”窗口。

用户的主要工作区域是“窗体设计器”，该窗口用来显示 Windows 窗体的“设计”视图（“Form1.cs[设计]”选项卡）或程序的“代码窗口”（Form1.cs 选项卡）。其他更多用于管理项目、调试程序等的子窗口，都可以通过在“视图”菜单中选择相应的命令将其打开。

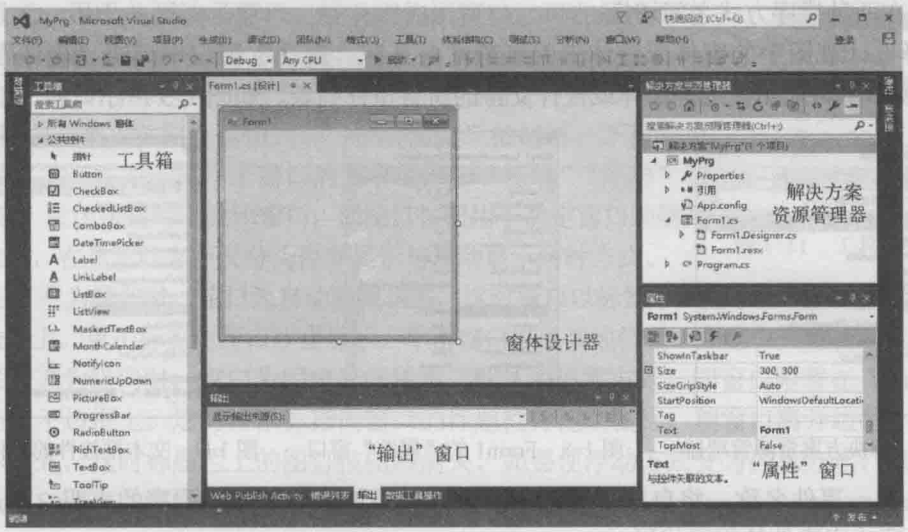

图 1-6　Visual Studio 2015 集成开发环境

在 Visual Studio 集成开发环境中，有两类子窗口，一类是在窗体设计器中显示的窗口，如“起始页”“代码窗口”和“设计”视图等；另一类是在窗体设计器周围显示的子窗口，如“工具箱”“解决方案资源管理器”“服务器资源管理器”“属性”“输出”和“错误列表”等。

1. 解决方案资源管理器

使用 Visual Studio 开发的应用系统称为一个解决方案，每一个解决方案可以包含一个或多个项目。一个项目通常是一个完整的程序模块或类模块，其中可能包含若干个文件。

“解决方案资源管理器”子窗口中显示了 Visual Studio 解决方案的树形结构，单击项目名称前面的空心或实心三角标记可以使该项展开或折叠。

如图 1-7 所示，在“解决方案资源管理器”窗口中可以像 Windows 资源管理器那样，浏览组成解决方案的所有项目和每个项目中的文件，可以对解决方案的各元素进行各种操作，如打开文件、添加内容、重命名和删除等。在“解决方案资源管理器”窗口中双击某个文件，将在主窗口中显示相应的视图（设计视图或代码窗口），以便对该文件进行编辑。

2. “属性”窗口

“属性”窗口用于设置解决方案中各对象的属性，当选择 Windows 窗体的设计视图、项目名称或类视图中的某一项时，“属性”窗口将以两列表格的形式显示该子项的所有属性。图 1-8 所示为在窗体设计器中选择了窗体控件 Form1 时显示的属性内容。

在“属性”窗口中，左边显示的是属性名称，右边显示的是对应各属性的属性值。选择某一名称后，可以在右边修改该属性值。例如，若希望修改在窗体的标题栏中显示的文字，可直接在“属性”窗口中修改其 Text 属性值。

在“属性”窗口的上方有一个下拉列表框，用于显示当前选定的对象名称及所属类型（如本例中 Form1 为对象名称，System.Windows.Forms.Form 为对象的类型）。可以在该下拉列

表框选择窗体中包含的其他对象。

如果选择的是窗体中的控件对象，在窗体的设计视图中，被选择对象会自动处于选定状态（四周出现 8 个控制点），原来选定的对象将取消选定。

属性名称默认按字母顺序排列，单击窗口中的“字母排序”按钮与“分类排序”按钮，可以在两种排序方式之间切换。

选择设计视图中的窗体或窗体中的某控件，并在“属性”窗口中单击“事件”按钮，“属性”窗口中将显示被选择窗体或控件支持的所有事件列表，如图 1-9 所示。

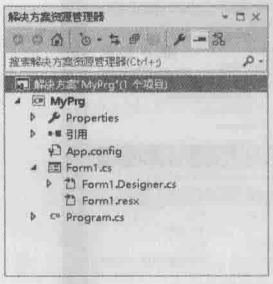

图 1-7　解决方案资源管理器

图 1-8　Form1 的“属性”窗口

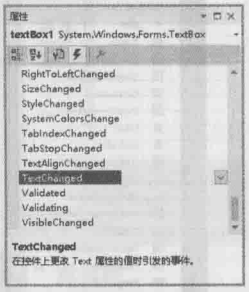

图 1-9　文本框控件的事件列表

双击某一事件名称，将自动打开代码窗口，并添加该事件处理程序的声明，用户可在其中填入处理响应事件的程序代码。

“属性”窗口的最下方有一个属性或事件功能的说明区域，当选择某一属性或事件时，说明区域会显示文字说明属性或事件的作用，这对初学者而言很有用。如果该区域没有显示，可将鼠标指向窗口列表框的下部边框，当鼠标指针变为双向箭头时向上拖动鼠标，该区域即可显示出来。

在事件子窗口中单击标题栏中的“属性”按钮，可返回“属性”窗口。

3. 工具箱

Visual Studio 工具箱中存放了众多系统预定义的、用于组成应用程序界面的“控件”，如文本框（TextBox）、按钮（Button）和标签（Label）等。

默认状态下，工具箱处于“自动隐藏”状态，窗口的左边框处显示有工具箱的选项卡标签。当鼠标指向该标签时，工具箱将显示到屏幕中。

工具箱用于向应用程序窗体中添加控件以构成应用程序的基本界面。利用工具箱使程序员可以像“拼图”一样，简单地设计程序界面。

如图 1-10 和图 1-11 所示，Visual Studio 将控件放在不同的分类中，各分类卡以空心三角符号表示折叠状态，以实心三角符号表示展开状态。默认情况下工具箱中的控件以名称的字母顺序排列，以方便用户查找控件。需要说明的是，工具箱中的内容只有在进入窗体设计视图时才会显示出来。

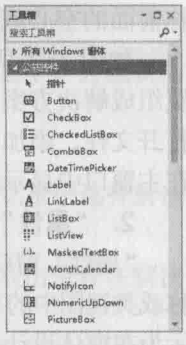

图 1-10　公共控件

如果进入了窗体的设计视图，工具箱仍没有出现在 Visual Studio 的 IDE 环境中，可选择“视图”→“工具箱”命令将其打开。

用户不但可以从工具箱中选择控件并将其拖动到窗体中，还可以将某一代码片断拖动到工具箱中暂存，以便将来重新使用。例如，可以将按钮（Button）控件从工具箱中拖放到窗体设计视图中，即向窗体中添加控件。也可从代码窗口中选择并拖出一个代码片段到工具箱中，待将来需要重复使用该代码段时，将其拖回代码窗口的适当位置即可。

4. 集成开发环境中子窗口的操作

如果在窗体设计器中显示的窗体或代码子窗口不止一个，则它们将会以选项卡的形式显示在窗体设计器的标题栏中，可以通过单击相应的选项卡标签进行切换。在这些选项卡的右侧都有一个“关闭”按钮×，用于关闭该子窗口。被关闭的子窗口，可在解决方案资源管理器中通过双击对应文件的方法将其再次打开。

所有子窗口的标题栏右侧都有 3 个操作按钮，一个“下拉菜单”按钮▾、一个“关闭”按钮×和一个“图钉”按钮📌。下拉菜单按钮中提供了一些关于操作子窗口的菜单项，如“停靠”“浮动”和“隐藏”等。关闭按钮用于关闭窗口，而图钉按钮用于决定窗口的隐藏与显示状态，在显示状态下又分为停靠显示与浮动显示两种方式。

图 1-11　工具箱中控件分类

当图钉📌为横向时，窗口为自动隐藏状态，这时窗口以标签形式显示在 Visual Studio 的左、右、下边框上。单击标签后窗口才会显示，单击窗口以外其他位置，则窗口又重新隐藏。

当图钉📌为纵向时，窗口为固定显示状态，默认为停靠方式，即窗口附着在 Visual Studio 的左、右、下边框上。这时将鼠标指向窗口的标题栏并拖动鼠标，使窗口离开边框，窗口即为浮动显示方式，这时标题栏上的图钉按钮将消失。如要使浮动方式变为停靠方式，只需拖动窗口至 Visual Studio 主窗口的边框上即可。

1.2.3　Visual Studio 的帮助系统

Visual Studio 是一个非常庞大的应用程序开发系统，其中涉及的概念、语法及函数自然很多，这使得程序员在工作时很难将需要的知识点准确无误地完全记忆下来。为了解决这一问题，Visual Studio 提供了一个完备的帮助系统。在 Visual Studio IDE 中，可以使用 F1 帮助、智能感知和 MSDN 资源等多种途径获取相关帮助信息。

1. 智能感知

在代码编写过程中，Visual Studio 提供了“智能感知”的帮助方式。利用这种帮助方式不仅可以提高代码输入效率，更重要的是避免了用户的输入错误。

（1）提示类、方法或对象名

在设计代码的过程中，当输入类名或对象名时，Visual Studio 会动态提供当前可用的类及对象列表，如图 1-12 所示。如果选择某一列表项，则在其右侧动态显示出简要说明。用户可通过双击某列表项完成自动输入，也可以用键盘的〈↑〉〈↓〉方向键选择所需选项后按〈Enter〉键。

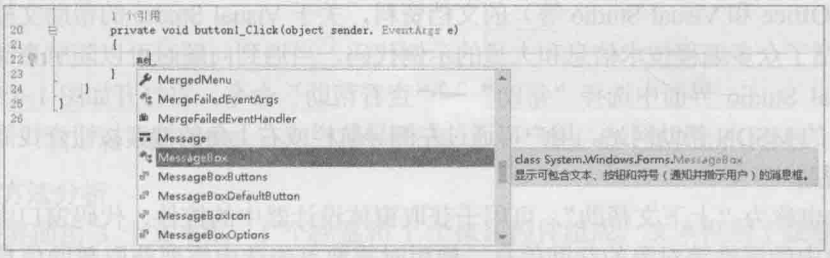

图 1-12　提示类名或对象名

C#代码是区分大小写的，使用智能提示就可以不用关心这一问题。例如，希望输入的类名称是 MessageBox，在输入时可以不考虑大小写直接输入首字母 m，通过观察提示列表中是

否出现了希望的方法名以决定是否继续输入后续字母“e”。

（2）提示类成员或对象成员

希望的类或对象名已出现在智能感知列表中，且已成为当前选中项，若此时需要继续输入类或对象成员时，可直接输入“.”号，系统将进一步提供新的可选项列表。例如，希望调用 MessageBox 类的 Show 方法显示一个提示信息框，则输入类的前两个字母 me 后，类名 MessageBox 成为当前选中项，此时直接输入“.”号，智能感知首先自动输入类名 MessageBox，而后动态显示该对象所具有的 3 个静态成员，如图 1-13 所示，再输入 Show 方法的首字母 s 后，该方法成为当前选中项。方法名右侧显示的是 Show 方法的语法格式提示。

需要特别注意的是，如果输入“.”号后没有出现正确的成员列表，很有可能是出现了输入错误。

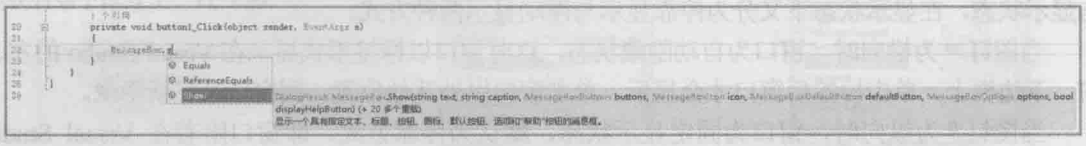

图 1-13　显示类成员列表

（3）提示方法的参数说明

当使用类的静态方法或成员方法时，Visual Studio 动态显示该方法的功能、不同用法（重载）及每种用法的参数说明。如图 1-14 所示，通过智能提示输入了 MessageBox.Show 后，输入后续的前括号“(”，系统将自动补全后括号“)”，并显示方法的第 1 个（共 21 个）参数书写格式。单击列表框中的“向下”按钮▼、“向上”按钮▲或使用键盘上的〈↓〉、〈↑〉键，可依次显示每种格式的具体用法和参数的类型及含义。

图 1-14　提示方法的语法格式及重载

（4）拼写检查

实时进行语法、关键字的拼写检查是智能提示的另一个十分有用的功能。用户在输入代码时若出现了拼写或格式错误，系统将以红色下画波浪线、绿色下画波浪线等形式给出提示。红色表示错误，绿色表示不严谨。用鼠标指向出现下画波浪线的地方，系统将显示详细的说明信息。

2. MSDN Library 帮助系统

MSDN Library（https://msdn.microsoft.com，以下简称 MSDN）提供了所有 Microsoft 产品（Windows、Office 和 Visual Studio 等）的文档资料，关于 Visual Studio 的帮助文档仅是其中一部分。它包括了众多编程技术信息和大量的示例代码，当遇到问题时可以随时查阅。

在 Visual Studio 界面中选择“帮助”→“查看帮助”命令，将打开如图 1-15 所示的关于 Visual Studio 的 MSDN 帮助网站。用户可通过左侧导航栏或右上角的搜索按钮查找需要的信息。

3. F1 帮助

F1 帮助也称为“上下文帮助”，可用于获取窗体设计器中的控件、代码窗口中的关键字和“属性”窗口中的属性等对象的帮助信息。使用时需要首先选中需要获取帮助信息的对象或将插入点光标定位到代码的关键字中后按〈F1〉键，系统将自动在 MSDN 中搜索，并将找到的相关帮助信息显示到屏幕中。