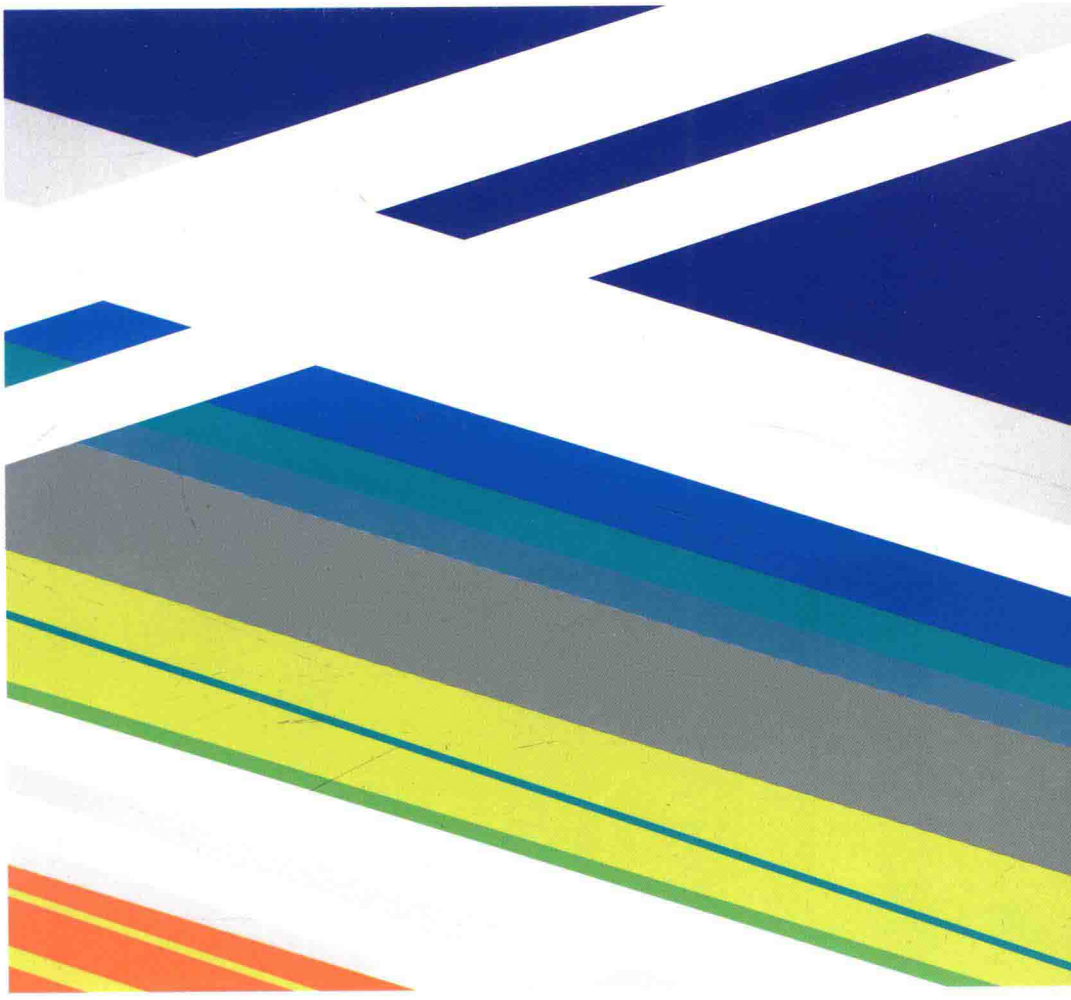

C#程序设计案例教程

主　编　郭树岩　刘一臻

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

C#程序设计案例教程

主　编　郭树岩　刘一臻
副主编　常沛鹏　胡　楠　王彦明

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

本书以 Microsoft Visual Studio 2008 为开发环境，通过 3 个生动有趣的实例培养学生的程序逻辑思维能力，完成 C#程序的入门学习；以 3 个实际项目为载体，从计算机专业人员在实际工作中所需的基础能力和技术出发，培养学生开发桌面型和中小 C/S 架构程序的职业能力和职业素养。本书对每个项目的每个功能模块都有详细的代码分析，让每一个初学者都能读懂，方便教与学。

本书主要覆盖的知识面包括：C# 3.0 语法、C#面向对象基础知识、C#控制台应用程序、Windows 基本控件的应用程序、GDI + 图形图像处理、ADO. NET 数据库访问技术等。本书可作为职业教育或应用技术型本科计算机专业程序入门类的项目导向性教材，也可作为 . NET（C#）培训班或认证培训用教材，还可供自学者参考使用。

图书在版编目（CIP）数据

C#程序设计案例教程 / 郭树岩，刘一臻主编 . —北京：北京理工大学出版社，2016. 12

ISBN 978 - 7 - 5640 - 9432 - 4

Ⅰ. ①C…　Ⅱ. ①郭…②刘…　Ⅲ. ①C 语言 - 程序设计 - 教材　Ⅳ. ①TP312. 8

中国版本图书馆 CIP 数据核字（2017）第 004154 号

出版发行 / 北京理工大学出版社有限责任公司
社　　址 / 北京市海淀区中关村南大街 5 号
邮　　编 / 100081
电　　话 / （010）68914775（总编室）
（010）82562903（教材售后服务热线）
（010）68948351（其他图书服务热线）
网　　址 / http：//www. bitpress. com. cn
经　　销 / 全国各地新华书店
印　　刷 / 三河市华骏印务包装有限公司
开　　本 / 787 毫米 × 1092 毫米　1/16
印　　张 / 17
字　　数 / 400 千字
版　　次 / 2016 年 12 月第 1 版　2016 年 12 月第 1 次印刷
定　　价 / 52. 00 元

责任编辑 / 王玲玲
文案编辑 / 王玲玲
责任校对 / 周瑞红
责任印制 / 李志强

前　言

C#是微软公司开发的一种面向对象的编程语言，是微软 . NET 开发环境的重要组成部分。而 Microsoft Visual C# 是微软公司开发的 C#编程集成开发环境，是为生成在 . NET Framework 上运行的多种应用程序而设计的。C#可以开发常见的 Web 应用程序和 Windows 应用程序，以其简单易用的编程界面以及高效的代码编写方式，深受广大编程人员的欢迎。

学习语言的目的是开发项目，但对于初学者，如何在学习基础知识后能独立开发项目，还存在一定的难度。本书就是为了帮助读者解决这个问题而编写的，本书所有项目都是 WinForm 项目，对每个项目都给出详细的代码，关键代码都给出解释并标注出来，便于教与学。

全书共分为 6 章，第 1 ~ 3 章主要介绍 C#的基本语法，每一章都列举若干实际程序，通过程序描述、实现步骤及代码分析获取相应知识点。其中，第 1 章 C#入门，通过实际程序让学生掌握 C#控制台程序和 Windows 程序的使用步骤，掌握 C#的基本语法，包括 C#数据类型、常量、变量及运算符的使用；第 2 章介绍 C#程序设计的流程控制语句，详细介绍选择语句、循环语句及数组的使用；第 3 章介绍面向对象程序设计的基础知识，包括类、构造函数、析构函数、方法、属性、类的继承和封装等；第 4 章介绍 Windows 应用程序开发，详细介绍常用的各种控件及其应用，并给出记事本项目的详细开发过程；第 5 章介绍图形图像编程（GDI + ）、Graphics 类、Pen 类和 Brush 类的使用，并给出 GDI + 图形编程项目的详细开发过程；第 6 章介绍 C#数据库的应用，探讨了如何使用 ADO. NET 进行数据访问，介绍了 ADO. NET 中的数据提供程序和 DataSet 对象，以及如何利用这些对象访问数据，此外，介绍了如何使用数据绑定技术实现数据的填充过程，并给出了通讯录项目的详细开发过程。

本书适合作为教材，主要面向那些希望学习 C#、没有面向对象概念而且缺乏开发经验的学生及初学者。与现有其他教材相比，本书从应用与工程实践的角度出发，重在激发学生的学习兴趣和将所学知识应用于程序开发实际能力的培养。通过本书的学习，可以达到以下目的：

（1）熟悉 Visual Studio. NET 开发、调试应用程序的步骤和方法。

（2）掌握 Windows 应用程序设计的方法和技巧。

（3）在项目中贯穿面向对象程序设计的思想，掌握使用 C#进行面向对象程序设计的方法。

（4）掌握常见图形应用程序的编写方法。

（5）掌握 ADO. NET 对象模型体系，学会使用多层架构搭建数据库应用程序。

本书由郭树岩、刘一臻主编；卢志鹏、胡楠、王彦明副主编。第 1 章和第 5 章由刘一

臻编写；第2章由卢志鹏编写；第3章由王彦明编写；第4章由郭树岩编写；第6章由胡楠编写。全书由郭树岩统稿。在编写本书过程中，编者参阅了大量的文献资料和网站资料，在此对所参考资料的作者也表示感谢。

本书文稿的录入，程序编写、运行以及插图的截取都是在Windows环境下同步进行的，所有程序都已在Visual Studio. NET 2008中文版环境中调试运行通过。由于编写时间和作者水平有限，书中不当之处在所难免，敬请广大读者批评指正。

编　者

CONTENTS 目录

第1章

C#入门

1.1 第一个控制台应用程序设计实例

1.1.1 程序描述

本例将创建一个简单却结构完整的C#控制台程序，即设计一个计算圆面积的控制台应用程序。

通过本示例，应学会：

①创建一个结构合理的控制台程序并运行调试；

②能够使用控制台输出函数WriteLine输出各种字符串及特殊字符。

1.1.2 实现步骤

在Visual Studio 2008（简称VS2008）中创建控制台应用程序的步骤如下：

1. 新建控制台项目

运行VS2008，执行“文件”→“新建”→“项目”菜单命令，弹出“新建项目”对话框，选择Visual C#的Windows项目类型，选择控制台应用程序模板，项目命名为“EX1_1”，如图1.1所示。

2. 添加代码

单击“确定”按钮后，系统新建了一个命名为“EX1_1”的控制台项目，并打开Program.cs文件。添加代码，代码如下：

```
using System;
using System.Collections.Generic;using System.Linq;
using System.Text;
namespace EX1_1                                    //定义的命名空间
{
    class Program                                  //定义类 Program
```

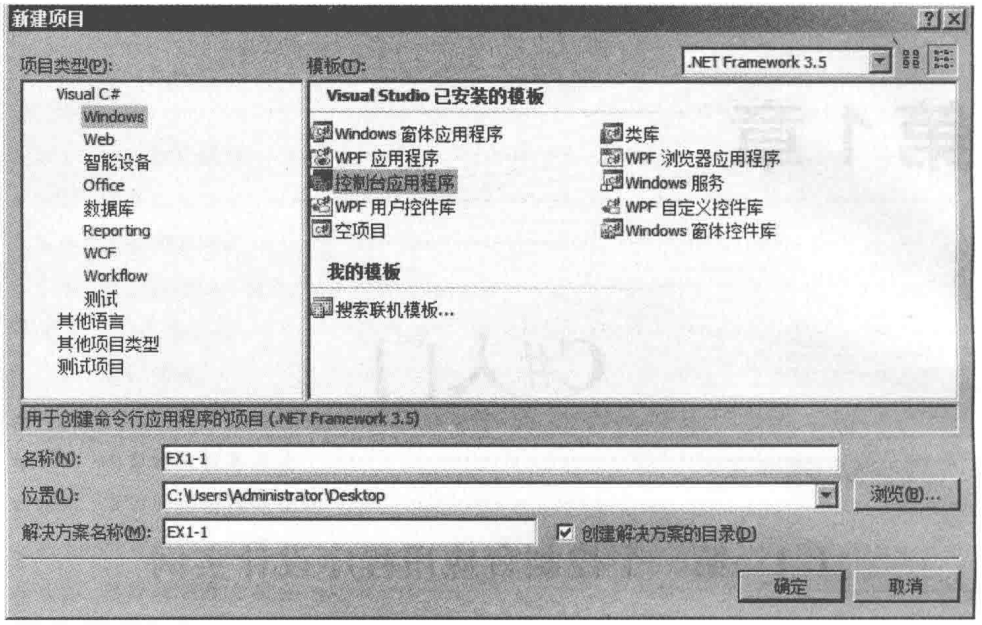

图 1.1

```
    {
        static void Main(string[]args)                    //程序的入口点
        {
            Console.WriteLine("请输入圆的半径");    /* 输出"请输入圆
                                                        的半径"提示字
                                                        样* /
            string r = Console.ReadLine();              /* 读入所输入的
                                                        字符* /
            double IntR = Convert.ToDouble(r);          /* 将字符类型转换
                                                        为数值类型* /
            const double PI = 3.1415926;                //定义圆周率
            Console.WriteLine(PI* IntR* IntR);          /* 计算圆面积并
                                                        输出* /
Console.ReadKey();
        }
    }
}
```

3. 运行程序，计算圆的面积

按 Ctrl + F5 快捷键或“调试”→“开始执行”运行程序，输入半径值“12”，结果如图 1.2 所示。

图 1.2

说明：

①命名空间提供了一种组织相关类和其他类型的方式，当引用了命名空间时，即可直接调用其中的类。例如，

System 是一个命名空间，Console 是该命名空间中的类。在后面章节中将详细介绍。

②Console 类属于 System 命名空间，表示控制台应用程序的标准输入、输出流和错误流。提供用于从控制台读取单个字符或整行的方法，还提供若干写入方法，可将值类型的实例、字符数组以及对象集自动转换为格式化或未格式化的字符串，然后将该字符串（可选择是否尾随一个行终止字符串）写入控制台。

③static void Main 为程序定义了入口点。应用程序启动时，Main 方法是第一个调用的方法，程序总是以 Main 函数后的一对花括号为开始和结束。一个 C#应用程序只能有一个入口点。

④WriteLine 方法将指定字符串显示到屏幕上，要显示的字符串用双引号（“”）括住。

⑤ReadKey 和 WriteLine 方法一样，都是 C#的标准类库的方法。将它放在这里，程序将会等待用户的输入，必须按下 Enter 键才能终止程序。这样就有时间查看程序运行的结果了。如果没有这句，则程序在执行完后将关闭，看到的将是闪了一下就关闭的控制台屏幕。

⑥在代码中“//”为代码注释符号，也可以使用“/* 所要注释内容 */”加以注释。如：

```
namespace EX1_1                    /* 定义的命名空间* /
```

⑦调试运行程序有两种方式，分别如下：

a. 执行“调试”→“开始执行（不调试）”菜单命令，或者直接按 Ctrl + F5 快捷键运行程序。这种方式只执行程序，并不调试程序。

b. 执行“调试”→“启动调试”菜单命令，或者直接按 F5 快捷键调试程序。这种方式需要设置断点，当程序执行到断点时，按 F10 快捷键逐步调试程序，也可以单击工具栏中的“▶”按钮启动调试。

1.1.3 注释及空白符的使用

1. 注释

注释是独立于代码的文档，不参与编译，是程序员用来交流想法的途径。注释通常反映程序员对代码逻辑的见解。因为程序可能会使用一段比较长的时间，并在这段时间内多次修改，需要修改时，程序员经常已经记不起特殊的细节，或者已经找不到原来的程序员了。这样从头去理解程序要花费大量的时间和精力。所以好的注释文档是相当重要的。

C#的注释有两种形式：

一种是多行注释：/* */，在/* 和 */之间的语句都不参加编译。

另一种是单行注释：//，即本行//后的语句为注释，不参与编译。

注释的作用主要有两点：

一是让程序员之间更好地交流。一般情况下，程序员习惯在程序的开头加上一段注释，标明该程序的基本信息。注释也经常用在一些较难理解的程序行后，起到解释的作用。

二是在调试程序时通过注释使一些不确定的代码不参加编译，以帮助程序员找出错误代码。

2. 空白符

C#程序使用空白符来分隔程序中使用的词和符号。空白符包括空格、制表符和换行符。

正确使用空白符可以提高程序的可读性。

C#程序中，单词之间必须用空白符来分隔。其他空白符都将被编译器忽略，不会影响到程序的编译和运行结果。但一个好的程序员应该养成合理使用缩进和对齐的好习惯，从而使程序的结构更加清晰。

1.1.4 Write 和 WriteLine 方法、字符串连接

1. Write 和 WriteLine 方法的基本应用

在该任务中，触发了如下 WriteLine 函数的语句：

```
Console.WriteLine(PI* IntR* IntR);
```

在这个语句中，Console 是 C#的控制台类；WriteLine 方法是 Console 类提供的一项服务，该服务的功能为在用户屏幕上输出表达式或字符串。可以说，把数据通过 WriteLine 方法发送消息给 Console，请求打印一些文本。

发送给 Console 方法的每个数据都称为参数（parameter）。在这个例子中，WriteLine 方法只使用了一个参数：要打印的表达式。

Console 类还提供了另一种可以使用的类似的服务：Write 方法。Write 方法和 WriteLine 方法的区别很小，但必须知道：WriteLine 方法打印发送给它的数据，然后光标移到下一行的开始；而 Write 方法完成后，光标则停留在打印字符串的末尾，不移动到下一行。例如：

```
Console.Write("老师");
Console.WriteLine("早晨好!");
Console.Write("我的启蒙老师");
Console.WriteLine();
Console.Write("-----谢谢您");
```

其运行结果如图 1.3 所示。

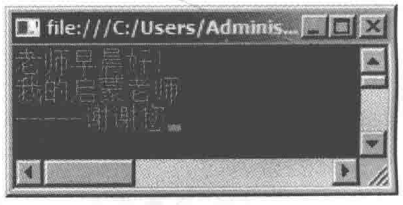

图 1.3

注意：WriteLine 方法是在打印完发送给它的数据后，才将光标移动到下一行的。

2. 字符串连接

由上述介绍可知，在程序中，代码是可以跨越多行的。因为编译器是以分语句结束标识的，回车换行不影响程序的编译。

但是，双引号中的字符串文字不能跨越多行！比如：下面的程序语句语法是不正确的，尝试编译时将会产生一个错误。

```
Console.WriteLine("你知道何时放假吗?
                  我还真是不知道。");
```

因此，如果想要在程序中打印一个比较长，无法在一行内写完的字符串，就可以使用字符串连接（string concatenation）将两个字符串头尾相连。字符串连接的运算符是加号（+）。例如，下面的表达式将两个字符串连接起来，产生一个较长的字符串：

```
Console.WriteLine("你知道何时放假吗?" +
"我还真是不知道。");
```

3. 转义序列

在C#中输出字符串时，双引号（"）用于指示一个字符串的开始和结束。假如想打印出一个引号，将它放在一对双引号中（" " "），编译器会感到困惑，因为它认为第2个引号是字符串的结束，而不知道该对第3个引号如何处理，结果导致一个编译错误。

C#语言定义了一些转义序列来表示特殊字符，见表1.1。转义字符由反斜杠（\）开始，它告诉编译器，后面跟的一个或者多个字符应该按照特殊的方式来解释编译。

表1.1

转义字符	意义
\'	用来表示单引号
\"	用来表示双引号
\ \	用来表示反斜杠
\0	用来表示空字符
\a	用来表示感叹号
\b	用来表示退格
\f	用来表示换页
\n	用来表示换行
\r	用来表示回车
\t	用来表示水平Tab
\v	用来表示垂直Tab

1.1.5 C#语言运行与调试

程序编写完后，就可以运行查看结果了。在Visual Studio 2008中，选择“调试”→“启动调试”，若程序没有语法错误，就能直接运行出结果；否则调试终止。启动调试的快捷键为F5。

此外，还可选择“调试”→“逐语句”，若程序没有语法错误，则Visual Studio 2008编译器将从Main函数开始，逐行执行代码，正在执行的代码行以黄底高亮显示。采用逐语句调试可以逐行查看代码运行过程的详细情况。当程序出现运行错误时（没有语法错误，但运行出来的结果和预计的不一样），也可以通过逐语句运行来帮助找出错误所在。逐语句调试的快捷键为F11。

在例EX1_1中设置断点逐步调试程序。

调试步骤如下：

①单击语句“Console.WriteLine（"请输入圆的半径");”的左端设置断点，如图1.4所示。

②按F5快捷键调试程序，程序执行到断点处停止执行，如图1.5所示。

③按F10快捷键逐步调试程序，当弹出控制台窗口提示输入圆的半径时，输入半径“5”，按Enter键继续运行程序，同时，在“自动窗口”中显示各个变量的值，最终将计算的结果显示在控制台窗口中。

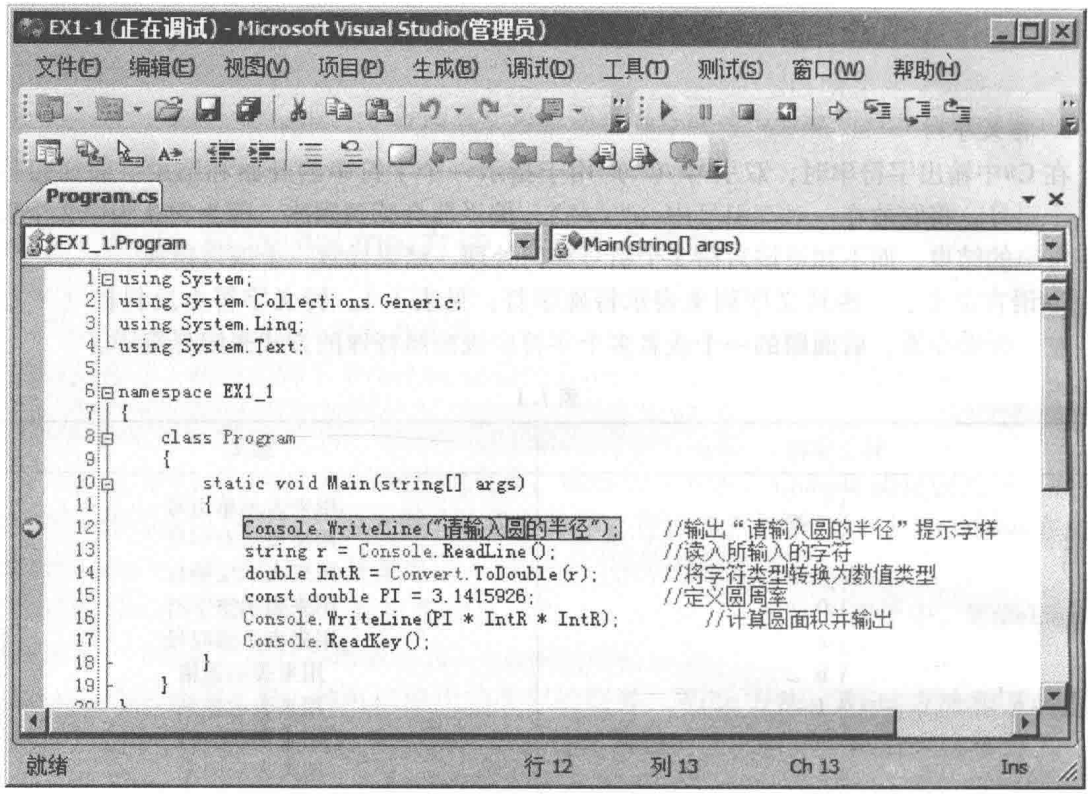
EX1-1 (正在调试) - Microsoft Visual Studio(管理员)
文件(F) 编辑(E) 视图(V) 项目(P) 生成(B) 调试(D) 工具(T) 测试(S) 窗口(W) 帮助(H)
Program.cs
EX1_1.Program
Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace EX1_1
{
class Program
{
static void Main(string[] args)
{
Console.WriteLine("请输入圆的半径"); //输出"请输入圆的半径"提示字样
string r = Console.ReadLine(); //读入所输入的字符
double IntR = Convert.ToDouble(r); //将字符类型转换为数值类型
const double PI = 3.1415926; //定义圆周率
Console.WriteLine(PI * IntR * IntR); //计算圆面积并输出
Console.ReadKey();
}
}
}
就绪
行 12
列 13
Ch 13
Ins

图 1.4

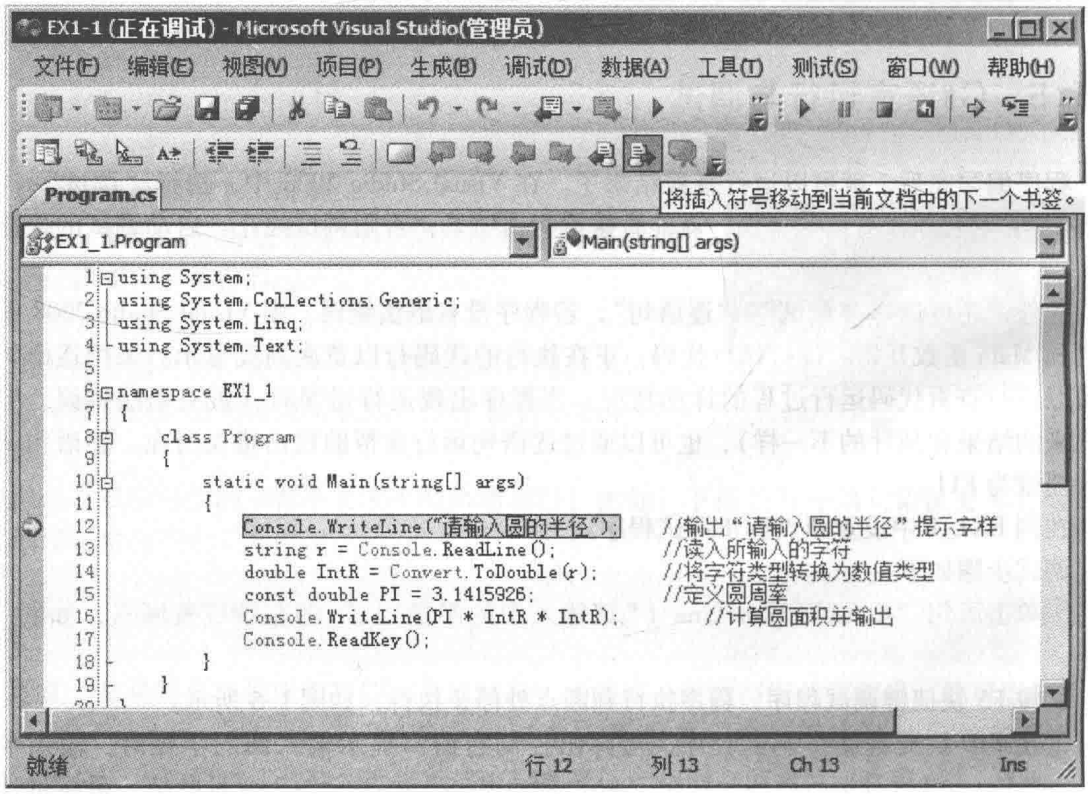
EX1-1 (正在调试) - Microsoft Visual Studio(管理员)
文件(F) 编辑(E) 视图(V) 项目(P) 生成(B) 调试(D) 数据(A) 工具(T) 测试(S) 窗口(W) 帮助(H)
Program.cs
将插入符号移动到当前文档中的下一个书签。
EX1_1.Program
Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace EX1_1
{
class Program
{
static void Main(string[] args)
{
Console.WriteLine("请输入圆的半径"); //输出"请输入圆的半径"提示字样
string r = Console.ReadLine(); //读入所输入的字符
double IntR = Convert.ToDouble(r); //将字符类型转换为数值类型
const double PI = 3.1415926; //定义圆周率
Console.WriteLine(PI * IntR * IntR); //计算圆面积并输出
Console.ReadKey();
}
}
}
就绪
行 12
列 13
Ch 13
Ins

图 1.5

1.2 第一个 Windows 应用程序设计实例

1.2.1 Visual Studio C# IDE 简介

Visual Studio C#集成开发环境（IDE）是一种通过常用用户界面公开的开发工具的集合。用户可以通过 IDE 中的窗口、菜单、属性页和向导与这些开发工具进行交互。打开 Visual Studio 2008，IDE 的默认外观如图 1.6 所示。

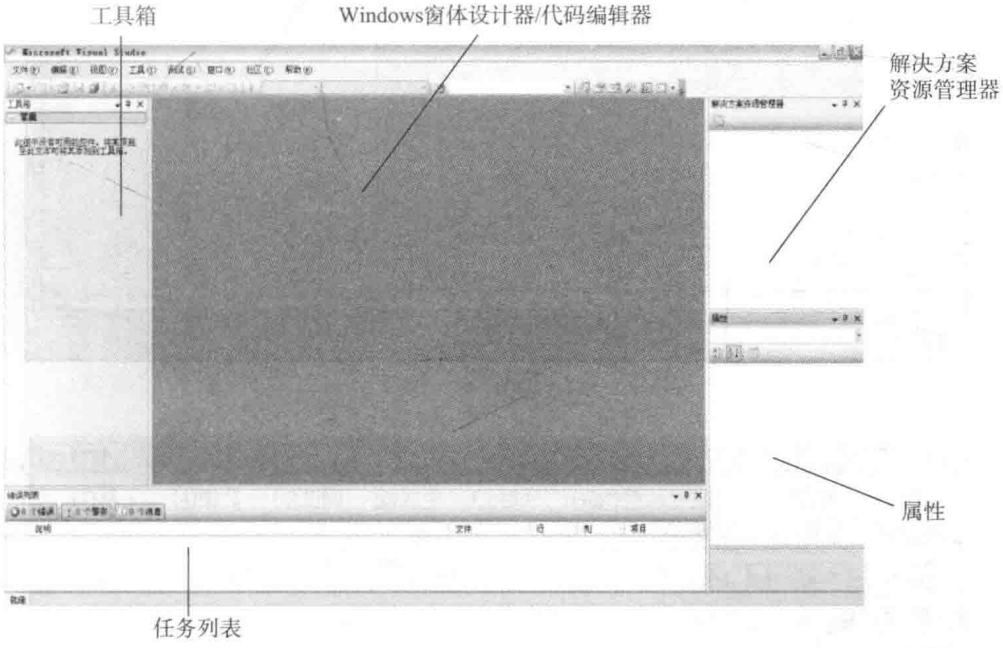

图 1.6

图 1.6 中所示工具的窗口都可从“视图”菜单打开。其中，较常用的有以下 4 种：

①代码编辑器：用于编写源代码。

②工具箱和 Windows 窗体设计器：用于使用鼠标迅速开发用户界面。

Windows 窗体设计器和代码编辑器在 IDE 的同一个位置上，可以通过上方的页面选择切换，如图 1.7 和图 1.8 所示。

③“属性”窗口：用于配置用户界面中控件的属性和事件，如图 1.9 所示。

④解决方案资源管理器：可用于查看和管理项目文件和设置。该窗口以分层树视图的方式显示项目中的所有文件。创建 Windows 窗体项目时，默认情况下，Visual C#会将一个窗体添加到项目中，并为其命名为 Form1，表示该窗体的两个文件称为 Form1. cs 和 Form1. Designer. cs。这些文件列表都可以在“解决方案资源管理器”中查看到，如图 1.10 所示。

图 1.10 中，Form1. cs 是写入代码的文件。

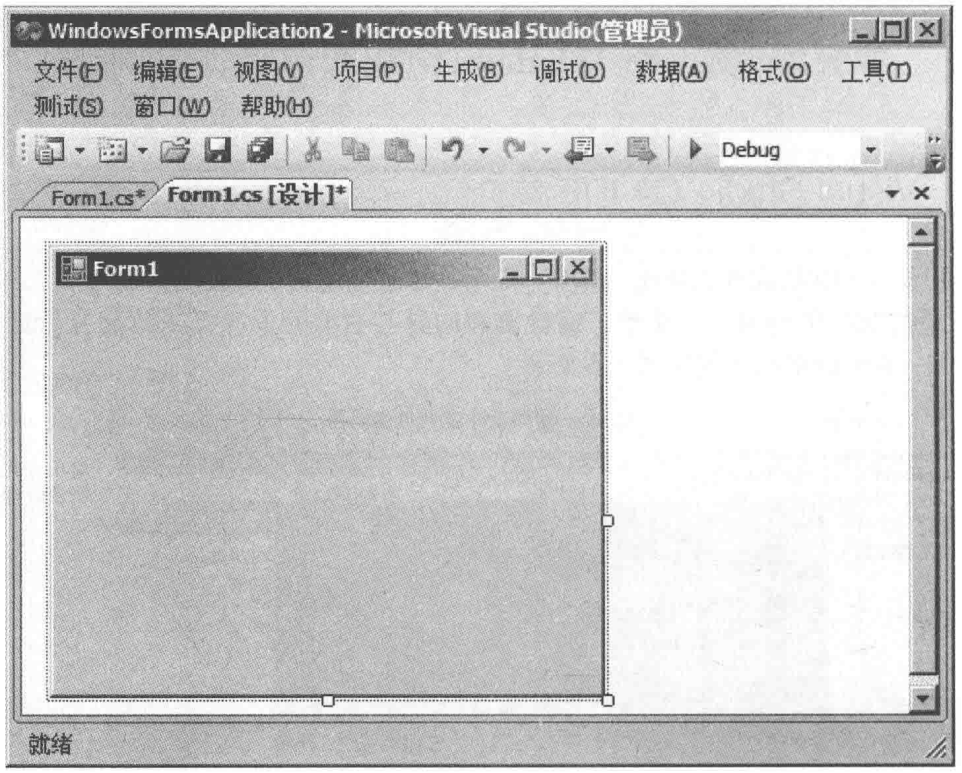

图 1.7

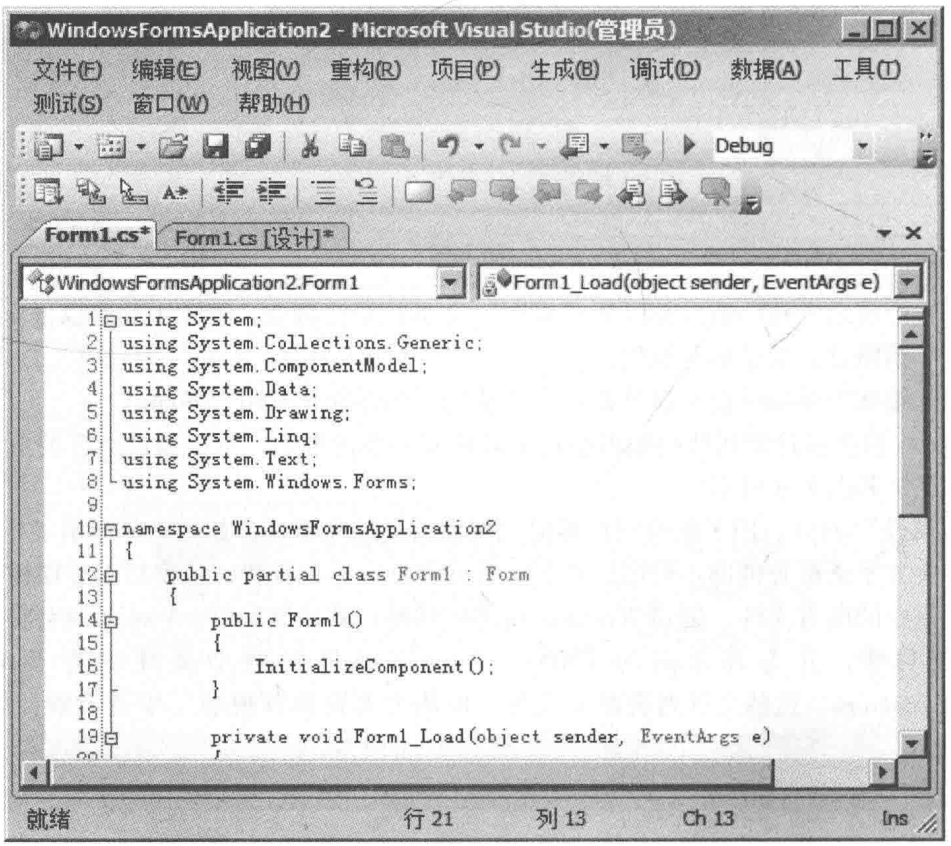

图 1.8