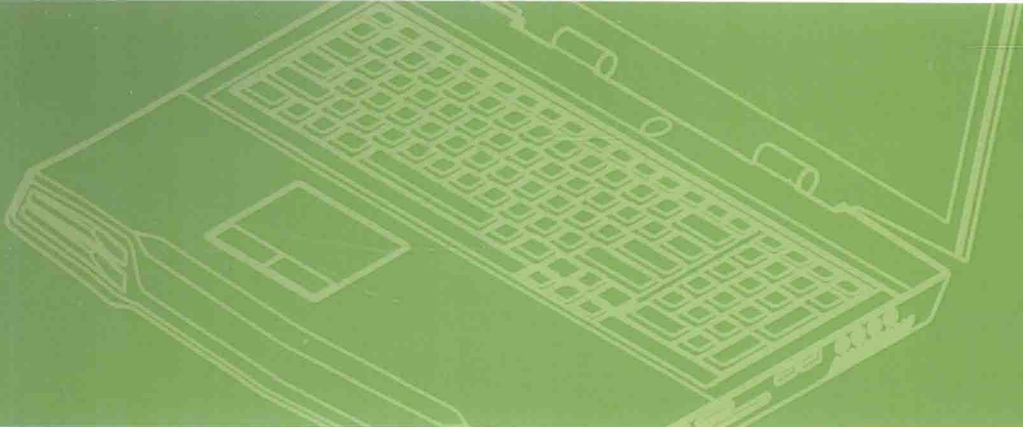
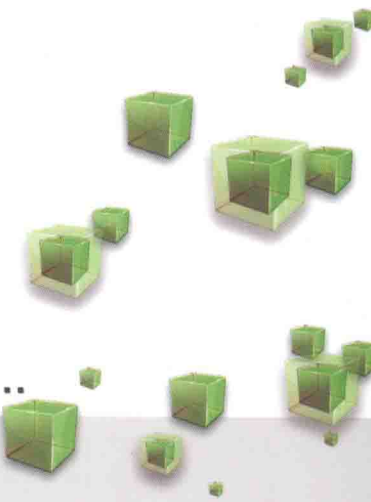




21世纪高等职业教育 计算机系列规划教材

# C#程序设计 与项目实战



◆ 庄 越 王槐彬 主 编  
◆ 周 江 关少珊 邹文杰 副主编



SOFTWARE



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>



配备  
电子课件

21 世纪高等职业教育计算机系列规划教材

# C#程序设计与项目实战

庄 越 王槐彬 主 编

周 江 关少珊 邹文杰 副主编

电子工业出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

本书介绍 C# 程序设计的基本语法、C# 面向对象程序设计、基于 Windows 的窗体应用程序开发、图形用户界面 GUI 设计、ADO.NET 数据库访问等内容。本书以 C# 语言规范 4.0 为标准, 开发工具使用 Visual Studio 2012, 内容选取上与工作岗位的能力要求紧密结合, 紧跟 C# 发展动向; 教材组织上以项目为载体, 校企合作精选工作项目, 任务案例实用精炼, 融知识性、趣味性于一体, 综合性和难度逐步提升, 符合学习规律, 也使读者在学习的过程中体会到 C# 编程的乐趣。本书配套建设了课程网站, 提供 PPT 教学课件、课程任务案例、实训资源包等的教学资源, 以方便教师教学和读者自主学习。

本教材职业特征鲜明, 逻辑清晰、形式生动活跃, 适合作为高职院校计算机专业学生相关课程的教材, 也可作为计算机编程爱好者的参考资料。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有, 侵权必究。

## 图书在版编目 (CIP) 数据

C# 程序设计与项目实战 / 庄越, 王槐彬主编. —北京: 电子工业出版社, 2014.6

21 世纪高等职业教育计算机系列规划教材

ISBN 978-7-121-23138-4

I. ①C… II. ①庄… ②王… III. ①C 语言—程序设计—高等职业教育—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2014) 第 090459 号

策划编辑: 徐建军 (xujj@phei.com.cn)

责任编辑: 郝黎明

印 刷: 北京市李史山胶印厂

装 订: 北京市李史山胶印厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1 092 1/16 印张: 20.25 字数: 518.4 千字

印 次: 2014 年 6 月第 1 次印刷

印 数: 3 000 册 定价: 38.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

# 前言

## Preface

C#（读作 C Sharp）是微软公司 .NET windows 网络框架的主角，是一种安全的、稳定的、简单的、优雅的，由 C 和 C++ 衍生出来的面向对象的编程语言。C# 集中了最近 20 年来软件开发技术积累的主要成果：面向对象、类型安全、组件技术、自动内存管理、跨平台异常处理、版本控制、代码安全管理等。C# 以其强大的操作能力，优雅的语法风格，创新的语言特性和便捷的面向组件编程的支持，迅速成为 Windows 平台上 .NET 开发的主角。

Microsoft Visual Studio（简称 VS）是目前最流行的 Windows 平台应用程序开发环境，其可视化设计器、智能代码助手、强劲的调试器等使程序员的工作快速而高效；解决方案管理器、良好的程序发布升级功能等使程序员可开发出企业级应用程序。本书采用版本为 Visual Studio 2012 版本，基于 .NET Framework 4.0。

本书根据高职“软件实现能力”的培养要求，基于工作过程导向的教学设计，以真实项目为载体，内容选取、教学重点与职业岗位的能力要求紧密结合。结合多年的教学实践，本书以 C# 语言规范 4.0 为标准，设计了 C# 程序设计的学习路径，包括以下五部分。

（1）C# 程序设计基础：第 1 章～第 5 章。读者将了解 C# 语言的特点，C# 程序设计的基本流程，C# 中常量和变量的定义与使用、数据类型与表达式，程序设计的三种基本结构，数组、字符串的处理，方法的使用等。

（2）面向对象程序设计：第 6 章和第 7 章。读者将学习面向对象的基本概念，包括类、委托、事件、类的继承、多态、接口等，掌握面向对象程序设计的基本方法。

（3）调试程序和异常处理：第 8 章。读者将学习使用 Visual Studio 2012 调试程序的方法，掌握 C# 应用程序中的异常处理技术，程序发布方法等。

（4）基于 Windows 程序设计：第 9 章～第 11 章。读者将学习 Windows 窗体应用程序开发、图形用户界面 GUI 设计、文件流操作等，能熟练使用控件设计基于 Windows 的窗体应用程序。

（5）数据库应用程序设计：第 12 章。读者将学习使用 ADO.NET 编程模型访问数据库，设计出高效、快捷的数据库应用程序。

本书由广东交通职业技术学院庄越、王槐彬担任主编，周江、关少珊、邹文杰担任副主编。此外，刘伯睿、李月、陈丽、支和才、徐燃柏、陈刚、官华、龙利国、李慧、曾小平等参加了本书的编写与校对工作。

本书所有程序均调试通过，同时为了方便教师教学，本书配有电子教学课件及相关资源，

请有此需要的读者可登录华信教育资源网 ([www.hxedu.com.cn](http://www.hxedu.com.cn)) 免费注册后进行下载, 如有问题可在网站留言板留言或与电子工业出版社联系 (E-mail:[hxedu@phei.com.cn](mailto:hxedu@phei.com.cn))。

虽然本书体现了我们近年 C#程序设计课程教学改革积累的经验, 但由于开发经验有限, 编写时间仓促, 书中难免存在疏漏和不足。恳请同行专家和读者能给予批评和指正。

编 者

# 目录

## Contents

|                          |      |
|--------------------------|------|
| 第 1 章 认知.NET 与 C#        | (1)  |
| 1.1 任务 1: 认知.NET         | (1)  |
| 1.1.1 .NET 发展历史          | (1)  |
| 1.1.2 .NET Framework     | (3)  |
| 1.1.3 程序编译               | (3)  |
| 1.1.4 C#                 | (4)  |
| 1.1.5 Visual Studio      | (5)  |
| 1.2 任务 2: 创建应用程序         | (11) |
| 1.2.1 基于控制台的应用程序         | (11) |
| 1.2.2 基于 Windows 窗体的应用程序 | (16) |
| 1.3 小结                   | (19) |
| 1.4 实训任务                 | (20) |
| 第 2 章 数据运算               | (21) |
| 2.1 任务 1: 标识符与关键字        | (21) |
| 2.1.1 命名标识符              | (22) |
| 2.1.2 关键字                | (22) |
| 2.2 任务 2: 使用常量和变量        | (22) |
| 2.2.1 常量                 | (22) |
| 2.2.2 变量                 | (23) |
| 2.3 任务 3: 认知基本数据类型       | (24) |
| 2.3.1 简单类型               | (25) |
| 2.3.2 枚举类型               | (27) |
| 2.3.3 结构类型               | (28) |
| 2.4 任务 4: 数据运算           | (29) |
| 2.4.1 运算符类型              | (29) |
| 2.4.2 赋值运算符与赋值表达式        | (29) |
| 2.4.3 算术运算符与算术表达式        | (30) |

|       |                        |      |
|-------|------------------------|------|
| 2.4.4 | 关系运算符与关系表达式 .....      | (31) |
| 2.4.5 | 逻辑运算符与逻辑表达式 .....      | (31) |
| 2.4.6 | 位运算符 .....             | (32) |
| 2.4.7 | 扩充后的赋值运算符 .....        | (33) |
| 2.4.8 | 条件运算符 .....            | (33) |
| 2.4.9 | 运算符优先级 .....           | (34) |
| 2.5   | 任务 5: 类型转换 .....       | (34) |
| 2.5.1 | 隐式转换 .....             | (35) |
| 2.5.2 | 显式转换 .....             | (36) |
| 2.4.3 | System.Convert 类 ..... | (36) |
| 2.4.4 | 数值和字符串之间的转换 .....      | (37) |
| 2.4.5 | 装箱和拆箱 .....            | (38) |
| 2.6   | 小结 .....               | (39) |
| 2.7   | 实训任务 .....             | (40) |
| 第 3 章 | 程序流程控制 .....           | (41) |
| 3.1   | 任务 1: 认知程序的三种结构 .....  | (41) |
| 3.1.1 | 顺序结构 .....             | (41) |
| 3.1.2 | 分支结构 .....             | (42) |
| 3.1.3 | 循环结构 .....             | (42) |
| 3.2   | 任务 2: 条件判断 .....       | (42) |
| 3.2.1 | if 语句 .....            | (42) |
| 3.2.2 | switch 语句 .....        | (45) |
| 3.3   | 任务 3: 循环嵌套 .....       | (47) |
| 3.3.1 | while 语句 .....         | (48) |
| 3.3.2 | do...while 语句 .....    | (49) |
| 3.3.3 | for 语句 .....           | (50) |
| 3.3.4 | foreach 语句 .....       | (54) |
| 3.4   | 流程跳转 .....             | (54) |
| 3.5   | 小结 .....               | (56) |
| 3.6   | 实训任务 .....             | (56) |
| 第 4 章 | 收集同一类型的数据 .....        | (59) |
| 4.1   | 任务 1: 创建和使用数组 .....    | (59) |
| 4.1.1 | 数组的概念 .....            | (59) |
| 4.1.2 | 声明和初始化数组 .....         | (60) |
| 4.1.3 | 访问数组 .....             | (62) |
| 4.1.4 | System.Array 类 .....   | (64) |
| 4.1.5 | 匿名数组 .....             | (65) |
| 4.2   | 任务 2: 集合处理 .....       | (66) |
| 4.2.1 | 非泛型集合 .....            | (66) |
| 4.2.2 | 泛型集合 .....             | (67) |

|                              |       |
|------------------------------|-------|
| 4.3 任务 3: 字符串处理 .....        | (68)  |
| 4.3.1 String 类 .....         | (68)  |
| 4.3.2 StringBuilder 类 .....  | (72)  |
| 4.4 小结 .....                 | (73)  |
| 4.5 实训任务 .....               | (74)  |
| <b>第 5 章 堆积木——方法</b> .....   | (77)  |
| 5.1 任务 1: 创建一个方法 .....       | (77)  |
| 5.2 任务 2: 调用方法 .....         | (78)  |
| 5.3 任务 3: 向方法传递参数 .....      | (82)  |
| 5.3.1 参数的类别 .....            | (82)  |
| 5.3.2 值参数调用 .....            | (82)  |
| 5.3.3 引用参数调用 .....           | (84)  |
| 5.4 任务 4: 方法的重载 .....        | (88)  |
| 5.5 任务 5: Main()方法 .....     | (89)  |
| 5.6 任务 6: 变量的作用域 .....       | (90)  |
| 5.6.1 局部变量 .....             | (91)  |
| 5.6.2 全局变量 .....             | (92)  |
| 5.7 小结 .....                 | (93)  |
| 5.8 实训任务 .....               | (93)  |
| <b>第 6 章 面向对象基础</b> .....    | (96)  |
| 6.1 任务 1: 认知面向对象 .....       | (96)  |
| 6.1.1 理解面向对象 .....           | (96)  |
| 6.1.2 面向对象的概念 .....          | (97)  |
| 6.2 任务 2: 面向对象程序设计 .....     | (98)  |
| 6.2.1 类 .....                | (98)  |
| 6.2.2 类成员 .....              | (99)  |
| 6.2.3 this 关键字 .....         | (111) |
| 6.2.4 委托 .....               | (113) |
| 6.2.5 事件 .....               | (116) |
| 6.3 小结 .....                 | (119) |
| 6.4 实训任务 .....               | (120) |
| <b>第 7 章 面向对象高级</b> .....    | (124) |
| 7.1 任务 1: 继承性 .....          | (124) |
| 7.1.1 理解继承 .....             | (124) |
| 7.1.2 继承的特性 .....            | (125) |
| 7.1.3 派生类 .....              | (125) |
| 7.1.4 尊老爱幼——派生类中调用构造函数 ..... | (127) |
| 7.1.5 密封类 .....              | (130) |
| 7.1.6 隐藏从基类继承的成员 .....       | (130) |
| 7.2 任务 2: 多态性 .....          | (133) |



|        |                          |       |
|--------|--------------------------|-------|
| 7.2.1  | 虚方法                      | (133) |
| 7.2.2  | 抽象方法和抽象类                 | (137) |
| 7.3    | 任务 3: 接口                 | (140) |
| 7.3.1  | 理解接口                     | (140) |
| 7.3.2  | 使用接口                     | (140) |
| 7.3.3  | 判别实现了某接口的对象              | (143) |
| 7.3.4  | 接口和抽象类的比较                | (144) |
| 7.4    | 小结                       | (145) |
| 7.5    | 实训任务                     | (145) |
| 第 8 章  | 调试程序和异常处理                | (149) |
| 8.1    | 任务 1: 调试程序               | (149) |
| 8.1.1  | 错误类型                     | (149) |
| 8.1.2  | 调试工具栏                    | (150) |
| 8.1.3  | 调试命令                     | (150) |
| 8.2    | 任务 2: 异常处理               | (154) |
| 8.2.1  | 异常类                      | (154) |
| 8.2.2  | 使用异常                     | (155) |
| 8.2.3  | 创建用户定义的异常类               | (159) |
| 8.3    | 任务 3: 生成程序               | (161) |
| 8.4    | 小结                       | (162) |
| 8.5    | 实训任务                     | (162) |
| 第 9 章  | 开发 Windows 窗体应用程序        | (164) |
| 9.1    | 任务 1: 认知 Windows 窗体应用程序  | (164) |
| 9.1.1  | 窗体                       | (165) |
| 9.1.2  | 控件                       | (165) |
| 9.1.3  | 事件                       | (165) |
| 9.2    | 任务 2: 认知 Windows 窗体      | (166) |
| 9.3    | 任务 3: 使用常用的 Windows 窗体控件 | (168) |
| 9.3.1  | Windows 窗体控件的共同特性        | (168) |
| 9.3.2  | 标签 Label                 | (170) |
| 9.3.3  | 文本框 TextBox              | (170) |
| 9.3.4  | 按钮 Button                | (172) |
| 9.3.5  | 单选按钮 RadioButton         | (174) |
| 9.3.6  | 复选框 CheckBox             | (176) |
| 9.3.7  | 列表框 ListBox              | (178) |
| 9.3.8  | 组合框 ComboBox             | (180) |
| 9.3.9  | 计时器 Timer                | (180) |
| 9.3.10 | 进度条 ProgressBar          | (180) |
| 9.3.11 | 日期时间 DateTimePicker      | (183) |
| 9.3.12 | 图片框 PictureBox           | (183) |

|               |                              |              |
|---------------|------------------------------|--------------|
| 9.3.13        | 图像列表组件 ImageList             | (184)        |
| 9.3.14        | 树视图 TreeView                 | (184)        |
| 9.3.15        | 列表视图 ListView                | (185)        |
| 9.3.16        | 消息框 MessageBox               | (186)        |
| 9.3.17        | 通用对话框                        | (188)        |
| 9.3.18        | 菜单 MenuStrip                 | (189)        |
| 9.3.18        | 工具栏 ToolStrip                | (195)        |
| 9.3.19        | 状态栏 StatusStrip              | (195)        |
| 9.4           | 任务 4: 构建多文档界面窗体              | (195)        |
| 9.4.1         | SDI 和 MDI 应用程序简介             | (195)        |
| 9.4.2         | 多文档界面特点                      | (196)        |
| 9.4.3         | 创建多文档界面                      | (196)        |
| 9.5           | 小结                           | (205)        |
| 9.6           | 实训任务                         | (205)        |
| <b>第 10 章</b> | <b>开发图形应用程序</b>              | <b>(208)</b> |
| 10.1          | 任务 1: 认知图形绘制基础               | (208)        |
| 10.1.1        | GDI+                         | (208)        |
| 10.1.2        | 图形坐标系统                       | (209)        |
| 10.1.3        | 绘图对象                         | (209)        |
| 10.2          | 任务 2: 绘制图形                   | (212)        |
| 10.2.1        | 绘图图面 Graphics                | (212)        |
| 10.2.2        | 画笔 Pen                       | (213)        |
| 10.2.3        | 画笔绘图                         | (214)        |
| 10.2.4        | 画刷 Brush                     | (220)        |
| 10.2.5        | 清理图面                         | (223)        |
| 10.3          | 任务 3: 绘制文本与图像                | (223)        |
| 10.3.1        | 文本                           | (223)        |
| 10.3.2        | 图像                           | (225)        |
| 10.4          | 任务 4: 交互绘图                   | (228)        |
| 10.4.1        | 鼠标事件                         | (228)        |
| 10.4.2        | 鼠标绘图                         | (229)        |
| 10.5          | 小结                           | (230)        |
| 10.6          | 实训任务                         | (231)        |
| <b>第 11 章</b> | <b>文件流操作</b>                 | <b>(232)</b> |
| 11.1          | 任务 1: 认知文件和流                 | (232)        |
| 11.1.1        | 文件和流的概念                      | (232)        |
| 11.1.2        | FileStream 类                 | (233)        |
| 11.2          | 任务 2: 操作文件和目录                | (235)        |
| 11.2.1        | File 类和 FileInfo 类           | (235)        |
| 11.2.2        | Directory 类和 DirectoryInfo 类 | (238)        |

|                              |              |
|------------------------------|--------------|
| 11.3 任务 3: 读写文件              | (240)        |
| 11.3.1 读写文本文件                | (240)        |
| 11.3.2 读写二进制文件               | (243)        |
| 11.4 小结                      | (246)        |
| 11.5 实训任务                    | (246)        |
| <b>第 12 章 数据库操作——ADO.NET</b> | <b>(249)</b> |
| 12.1 任务 1: 理解数据访问模型          | (251)        |
| 12.2 任务 2: 认知 ADO.NET 对象模型   | (252)        |
| 12.2.1 ADO.NET 对象模型          | (252)        |
| 12.2.2 数据提供程序                | (252)        |
| 12.2.3 DataSet               | (253)        |
| 12.2.4 ADO.NET 的数据访问模式       | (253)        |
| 12.3 任务 3: 创建数据库连接           | (254)        |
| 12.3.1 连接对象 Connection       | (254)        |
| 12.3.2 SqlConnection 属性      | (256)        |
| 12.3.3 应用程序配置文件              | (257)        |
| 12.3.4 管理数据库连接               | (258)        |
| 12.3.5 处理连接的异常               | (259)        |
| 12.4 任务 4: 在连接环境中获取数据        | (259)        |
| 12.4.1 命令对象 Command          | (259)        |
| 12.4.2 Command 对象的属性         | (261)        |
| 12.4.3 设置 Command 对象的属性      | (262)        |
| 12.4.4 Command 对象的参数         | (262)        |
| 12.4.5 使用 Command 对象执行命令     | (263)        |
| 12.5 任务 5: 断开连接的数据访问         | (274)        |
| 12.5.1 数据集 DataSet           | (274)        |
| 12.5.2 数据适配器 DataAdapter     | (275)        |
| 12.5.3 数据更新                  | (278)        |
| 12.6 任务 6: 创建三层体系结构的应用程序     | (294)        |
| 12.7 小结                      | (302)        |
| 12.8 实训任务                    | (302)        |
| <b>附录 A 实训报告书模板</b>          | <b>(310)</b> |
| <b>参考文献</b>                  | <b>(311)</b> |

# 第1章

## 认知.NET 与 C#

近年来软件技术在飞速发展。.NET Framework 是微软为开发应用程序创建的一个富有革命性的新平台。经过十年的发展,.NET Framework 已经发行了多个版本。如今,.NET Framework 不再是“吴下阿蒙”,已成为新一代企业级应用开发平台。

### ● 工作任务

- 任务 1: 认知.NET
- 任务 2: 创建应用程序

### ● 学习目标

- 熟悉 Visual Studio 集成开发环境 (IDE)
- 学会创建、编译和执行简单的.NET 应用程序
- 使用输入和输出语句
- 初步认识和了解窗体、控件、事件和方法

## 1.1 任务 1: 认知.NET

### ● 相关知识与技能

#### 1.1.1 .NET 发展历史

.NET 是微软公司的新战略,它包含微软公司对未来的核心战略、规划和洞察力,微软公司所有的产品都围绕这个战略开发。此战略的核心就是.NET Framework,该框架提供了全面支持.NET 的核心技术。

.NET 用来解决编程人员面临的许多问题:

- 它负责处理在创建大型、可靠的应用程序时的大量艰辛工作。

- 它允许程序员统一两种架构——在本地机器上运行的应用程序和通过 Web 访问的应用程序。
- 它减少了与编程框架相关的传统开销——不再需要高性能编程语言来编写复杂的代码以获取高速的.NET 程序。
- 它允许不同语言的程序员在同一个应用程序中协同工作。
- 它开始兼容各种最终用户工具，包括桌面、PDA 和手机等。

总之，.NET 提供了一种更简单、更快捷、更廉价的方式来获得高效的程序。

在某些方面，.NET 很像 Java。实际上，Java 的口号“一旦编写出来，就能在任何地方运行”也很符合.NET 的原则。Java 是“一种语言，多个平台”，而.NET 是“多个语言，一个平台”。

.NET 平台的架构图如图 1-1 所示。



图 1-1 .NET 平台框架图

微软.NET 的发展历史如下：

- 1998 年 12 月，微软公司提出.NET 计划后，开始开发.NET 平台上的面向对象程序的设计语言。
- 2000 年 2 月，微软公司正式将这种语言命名为 C#，读成“C Sharp”。据说命名为 C# 是因为开发小组的人很讨厌当时的搜索引擎，因此把大部分搜索引擎无法识别的“#”字符作为该语言名字的一部分；还有一种说法是因为音乐中“#”字符是升调记号，表达了微软公司希望 C#在 C 的基础上更上一层楼的美好愿望。
- 2000 年 7 月，微软公司发布了 C#语言的第 1 个预览版。
- 2002 年 2 月，微软公司发布了.NET 平台“Visual Studio.NET 2002”和 C#语言的第 1 个正式版本——C# 1.0。
- 2003 年 4 月，微软公司推出了 Visual Studio .NET 2003，同时也发布了.NET Framework 1.1 和 C#语言规范 1.2——C# 1.2。
- 2005 年 10 月 27 日，Visual Studio 2005、.NET Framework 2.0 和 C#语言规范 2.0(C# 2.0) 正式发布，在新的版本中提供了更多的类库和使用功能。
- 2007 年 8 月 20 日，继.NET Framework 3.0 发布后，C#语言规范 3.0 (C# 3.0) 发布。
- 2007 年 11 月 19 日，微软公司发布了 Visual Studio 2008 和.NET Framework 3.5。
- 2010 年 4 月 12 日，微软公司发布了 C#语言规范 4.0(C# 4.0)、Visual Studio 2010 和.NET Framework 4.0。
- 2012 年 9 月 12 日，微软公司发布了 Visual Studio 2012 和.NET Framework 4.5。

Visual Studio 是一种集成开发环境 (IDE)，它使开发人员能够在其中用多种语言 (包括 Visual C#) 之一来创建用于 .NET Framework 的程序。Visual Studio 从诞生到现在已经十年多了，Visual Studio 与 .NET Framework 的对应关系如表 1-1 所示：

表 1-1 Visual Studio 与 .NET Framework 的对应关系表

| Visual Studio 版本   | .NET Framework 版本  |
|--------------------|--------------------|
| Visual Studio 2002 | .NET Framework 1.0 |
| Visual Studio 2003 | .NET Framework 1.1 |
| Visual Studio 2005 | .NET Framework 2.0 |
| Visual Studio 2008 | .NET Framework 3.5 |
| Visual Studio 2010 | .NET Framework 4.0 |
| Visual Studio 2012 | .NET Framework 4.5 |

### 1.1.2 .NET Framework

.NET Framework (又称.NET 框架) 是用于代码编译和执行的集成托管环境。换句话说，它管理着应用程序运行的方方面面。所有的 .NET 应用程序都在 .NET Framework 上执行，受 .NET Framework 的管理。因此，.NET 应用程序并不直接和操作系统打交道，而是通过集成托管环境进行操作。

同时，.NET Framework 也提供了丰富的类库，供 .NET Framework 应用程序进行调用。.NET 应用程序中所调用的类库来自于 .NET Framework，编译后的程序运行在 .NET Framework 之上。

.NET Framework 由两个主要组件组成：

- 公共语言运行库 (Common Language Runtime, CLR)

公共语言运行库可视为管理代码执行的环境。它介于操作系统和应用程序之间，提供了代码编译、内存分配、线程管理以及垃圾回收之类的核心服务。它还强制实施了严格的类型安全性，并通过强制实施代码访问安全性来确保代码在安全的环境中执行。

- .NET Framework 类库

.NET Framework 类库提供了一套通用功能的标准代码，这些代码包含了开发人员可用于简化开发工作的类和其他类型的库。

### 1.1.3 程序编译

随着计算机技术的发展，计算机编程语言由最初的机器语言，逐渐演变为汇编语言，再最后发展为高级语言，人们逐渐使用类似英语的单词来编写代码。但计算机仅能够识别和执行以二进制代码形式表示的机器语言，这种高级语言编写的程序需要编译或解释成为机器语言才能在计算机上运行。

从高级语言编写的程序到低级语言编写的程序的转换有两种方法：解释或者编译。

- 解释：执行方式类似于我们日常生活中的“同声翻译”，应用程序源代码一边由相应语言的解释器“解释”成目标代码（机器语言），一边执行，因此效率比较低，而且不能生成可独立执行的可执行文件，应用程序不能脱离其解释器，但这种方式比较灵活，可以动态地调整、修改应用程序。

- 编译：编译是指在源程序执行之前，就将程序源代码“翻译”成目标代码（机器语言），因此其目标程序可以脱离其语言环境独立执行，使用比较方便、效率较高。但应用程序一旦需要修改，必须先修改源代码，再重新编译生成新的目标文件才能执行。只有目标文件而没有源代码，修改很不方便。现在大多数的编程语言都是编译型的，例如 C#、C++ 等。

C#语言编写的源程序扩展名为.cs。如果计算机上已经安装了.NET Framework，你可以使用.NET 命令行窗口，运行命令 `csc xxx.cs` 来编译刚写好的程序（xxx 表示源程序文件名）。此时会在存放 xxx.cs 文件的同一目录下出现一个名为 xxx.exe 的程序。通过双击此 xxx.exe 文件名，可以执行程序。

.NET 应用程序在编译时采用了两个步骤来解决代码和机器的交互问题。首先，.NET 编译器将程序代码编译成称为 MS 中间语言（MS Intermediate Language, MSIL）的特殊格式，再将其传给 CLR（公共语言运行库），CLR 使用 JIT（Just-In-Time）编译器将代码编译成真正的机器语言，并对程序进行最后的且与机器相匹配的优化，以使程序能在其所在的计算机上以尽可能快的速度运行。如图 1-2 所示。

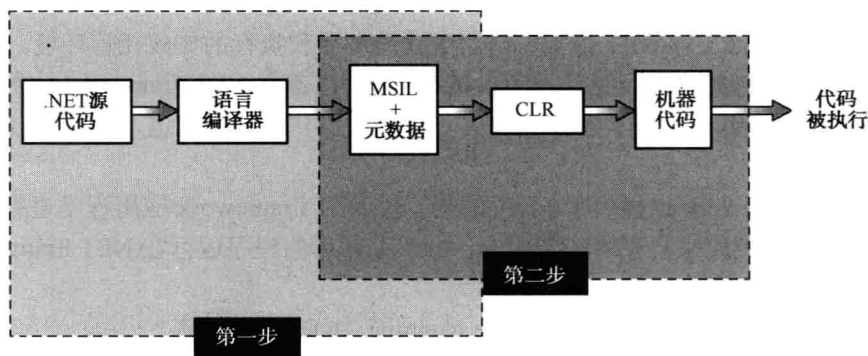


图 1-2 .NET 程序编译步骤

通过 MSIL 和 CLR 的组合使用，带来了前述两种代码转换的综合性优点，即获得了解释代码的可移植性和编译代码的结构优化特性。

更为重要的是，MSIL 本身是与机器无关的。因此，可以在装有 CLR 的任何一台计算机上运行。实际上，一旦编写出 .NET 程序代码并将其编译，就可以将它复制到装有 CLR 的任何计算机，并在该计算机上执行。

MSIL 可以由任何遵循 CLS（公共语言规范）的可读语言生成。另外，MSIL 编译器还支持其他 20 多种语言。因此，可以在应用程序内部交替地使用这些兼容语言，一旦将一套文件编译成 MSIL，它们都将统一为一种语言。这种灵活性允许不同的小组在同一个 Web 站点上用不同的语言协同工作。

#### 1.1.4 C#

.NET Framework 运行环境支持多种编程语言：C#、Visual Basic.NET、Visual C++.NET、J# 和 JScript.NET，但是 C# 是为它量身定做的，可以充分发挥 .NET Framework 的优势。C# 是微软公司开发的一种面向对象编程语言，它结合了 C++ 的强大功能和 Visual Basic 的易用性，而



且很多方面和 Java 类似。这些特点都使得使用 C# 开发程序更加灵活。

C# 是一种简单、灵活而又功能强大的程序设计语言，它使得程序员能够快速开发多种应用程序。由于它彻底采用了面向对象程序设计思想，因此无论是最普通的应用程序还是大型的商业软件，它都是最合适的选择。使用 C# 设计的组件能够轻松地转换成 XML Web Service，并且可以被互联网上的程序调用。

简而言之，C# 有以下 4 个特点：

#### (1) 简单

C# 继承了 C 和 C++ 的优点，并在此基础上进行了改善，使得语言更加简单。C# 同时摒弃了其他编程语言（例如 C++ 和 Java）中的一些复杂性和缺陷，这使得没有编程基础的程序员也可以有效地减少开发过程中的错误。

#### (2) 面向对象

C# 具有面向对象程序设计语言所应有的一切特性：封装、继承和多态。通过面向对象的强大功能，C# 使得程序员的编程效率得到极大的提高，缩短了应用程序的开发周期。

#### (3) 与 Web 紧密结合

C# 对于网络中结构化数据传输的标准——XML 提供了很好的支持，程序员能够利用简单的 C# 语言结构方便地开发 XML Web Service，有效地处理网络中的各种数据。

#### (4) 基于 .NET Framework

.NET Framework 为用 C# 编写的应用程序提供了安全性保障和错误处理机制。

### 1.1.5 Visual Studio

开发人员可以使用 Windows 自带的“记事本”来进行程序的开发。但是，如果所有的代码都要手动添加和输入，然后手动编译程序，使用命令调试程序等，那么工作效率是很低的。为了解决上述问题，集成开发环境应运而生。集成开发环境集成了方便程序员开发的各种工具和功能，如防止程序员犯低级错误或者笔误的自动语法检查、方便的图形化调试功能等，大大减少了编码的语法错误并降低了代价，使得程序的运行更简单。微软在推出 C# 这种开发语言的同时，也推出了对应的集成开发环境 Visual Studio 系列产品。

Visual Studio 2012 有多个版本，常用的有：Visual Studio Ultimate 2012、Visual Studio Premium 2012、Visual Studio Professional 2012、Visual Studio Test Professional 2012、Visual Studio Team Foundation Server 2012。

Visual Studio 2012 安装要求：支持 Windows 7 SP1 (x86 和 x64)、Windows 8 (x86 和 x64)、Windows Server 2008 R2 (x64)、Windows Server 2012 (x64) 操作系统；硬件要求：1.6 GHz 或更快的处理器、1 GB RAM（在虚拟机上运行时需要 1.5 GB）、10 GB (NTFS) 可用硬盘、5400 RPM 硬盘驱动器、支持 DirectX 9 的显卡，显示器分辨率大于或等于 1024×768。

## ● 任务实施与总结

### 1. 使用 Visual Studio 2012

在 Windows 状态下，选择“开始”|“所有程序”|“Microsoft Visual Studio 2012”|“Microsoft Visual Studio 2012”命令，进入 Visual Studio 2012 系统起始页，如图 1-3 所示。





图 1-3 Visual Studio 2012 起始页

主窗口的标题栏、菜单栏和工具栏位于集成开发环境用户界面的顶部。

#### (1) 标题栏

标题栏就是窗口顶部的水平条，显示出当前激活的项目名和应用软件名称。

#### (2) 菜单栏

在创建应用程序后，菜单栏中显示“文件(F)”、“编辑(E)”、“视图(V)”、“项目(P)”、“生成(B)”、“调试(D)”、“团队(M)”、“SQL(Q)”、“工具(T)”、“测试(S)”、“分析(N)”、“窗口(W)”和“帮助(H)”等菜单项，如图 1-4 所示。

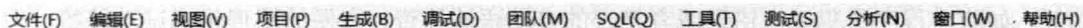


图 1-4 菜单栏

菜单各项的主要功能说明如下：

##### ● “文件(F)”菜单

用于新建、打开、保存或显示最近的项目。例如，选择“新建项目”子菜单项可以建立一个新项目；选择“新建网站”子菜单项将建立一个 Web 网站；选择“打开项目”、“打开网站”、“打开文件”将分别用于打开本地的或网络中的项目、网站或文件；选择“全部保存”子菜单将保存当前打开的所有文件等。

##### ● “编辑(E)”菜单

用于程序源代码编辑。编辑菜单可以剪切、拷贝、粘贴、选择、查找和替换文字等。

##### ● “视图(V)”菜单

视图菜单可以打开各种窗口供用户使用。例如选择“解决方案资源管理器”子菜单将显示“解决方案资源管理器”窗口。选择“属性窗口”子菜单显示窗体或控件对象的“属性”窗口。选择“错误列表”子菜单将在下部显示“错误列表”窗口。选择“工具箱”子菜单将显示“工具箱”窗口等。