

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C#程序设计及应用教程 (第2版)

C# Programming and Applications (2nd Edition)

马骏 主编 邓居英 杨湖 副主编

- 内容全面: C#程序设计 + Windows程序设计 + Web程序设计
- 实用性强: 应用实例 + 使用技巧 + 每章实践练习题
- 配套丰富: PPT + 实例源程序代码 + 习题参考答案



精品系列



人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C#程序设计及应用教程 (第2版)

C# Programming and Applications (2nd Edition)

马骏 主编 邓居英 杨湖 副主编



精品系列

人民邮电出版社

北京

图书在版编目 (CIP) 数据

C#程序设计及应用教程 / 马骏主编. —2版. —北京: 人民邮电出版社, 2009. 5
21世纪高等学校计算机规划教材
ISBN 978-7-115-19825-9

I. C… II. 马… III. C语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆CIP数据核字 (2009) 第033656号

内 容 提 要

本书详细介绍 C#程序设计及其应用技术。全书由 C#程序设计基础、Windows 应用程序设计和 Web 应用程序设计三部分组成, 主要内容包括 C#语言基础知识、面向对象的编程技术、Windows 窗体控件、目录与文件管理、图形图像处理、ADO.NET、报表设计、类库与控件库设计以及 Web 应用程序开发。

本书可作为高等院校计算机及相关专业的教材, 也可作为初、中级程序员的参考用书。

21 世纪高等学校计算机规划教材

C#程序设计及应用教程 (第 2 版)

- ◆ 主 编 马 骏
副 主 编 邓居英 杨 湖
责任编辑 邹文波
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京鑫正大印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
印张: 22.5
字数: 588 千字
印数: 12 001—15 000 册
- 2009 年 5 月第 2 版
2009 年 5 月北京第 1 次印刷

ISBN 978-7-115-19825-9/TP

定价: 35.00 元

读者服务热线: (010)67170985 印装质量热线: (010)67129223

反盗版热线: (010)67171154

出版者的话

计算机应用能力已经成为社会各行业从业人员最重要的工作技能要求之一，而计算机教材质量的好坏会直接影响人才素质的培养。目前，计算机教材出版市场百花争艳，品种急剧增多，要从林林总总的教材中挑选一本适合课程设置要求、满足教学实际需要的教材，难度越来越大。

人民邮电出版社作为一家以计算机、通信、电子信息类图书与教材出版为主的科技教育类出版社，在计算机教材领域已经出版了多套计算机系列教材。在各套系列教材中涌现出了一批被广大一线授课教师选用、深受广大师生好评的优秀教材。老师们希望我社能有更多的优秀教材集中地呈现在老师和读者面前，为此我社组织了这套“21世纪高等学校计算机规划教材——精品系列”。

本套教材具有下列特点。

(1) 前期调研充分，适合实际教学需要。本套教材主要面向普通本科院校的学生编写，在内容深度、系统结构、案例选择、编写方法等方面进行了深入细致的调研，目的是在教材编写之前充分了解实际教学的需要。

(2) 编写目标明确，读者对象针对性强。每一本教材在编写之前都明确了该教材的读者对象和适用范围，即明确面向的读者是计算机专业、非计算机理工类专业还是文科类专业的学生，尽量符合目前普通高等教育计算机课程的教学计划、教学大纲以及发展趋势。

(3) 精选作者，保证质量。本套教材的作者，既有来自院校的一线授课老师，也有来自IT企业、科研机构等单位的资深技术人员。通过他们的合作使老师丰富的实际教学经验与技术人员丰富的实践工程经验相融合，为广大师生编写出适合目前教学实际需求、满足学校新时期人才培养模式的高质量教材。

(4) 一纲多本，适应面宽。在本套教材中，我们根据目前教学的实际情况，做到“一纲多本”，即根据院校已学课程和后续课程的不同开设情况，为同一科目提供不同类型的教材。

(5) 突出能力培养，适应人才市场要求。本套教材贴近市场对于计算机人才的能力要求，注重理论知识与实际应用的结合，注重实际操作和实践动手能力的培养，为学生快速适应企业实际需求做好准备。

(6) 配套服务完善。对于每一本教材，我们在教材出版的同时，都将提供完备的PPT课件，并根据需要提供书中的源程序代码、习题答案、教学大纲等内容，部分教材还将在作者的配合下，提供疑难解答、教学交流等服务。

在本套教材的策划组织过程中，我们获得了来自清华大学、北京大学、中国人民大学、浙江大学、吉林大学、武汉大学、哈尔滨工业大学、东南大学、四川大学、上海交通大学、西安交通大学、电子科技大学、西安电子科技大学、北京邮电大学、北京林业大学等院校老师的大力支持和帮助，同时获得了来自信息产业部电信研究院、联想、华为、中兴、同方、爱立信、摩托罗拉等企业和科研单位的领导或技术人员的积极配合。在此，向他们表示衷心的感谢。

我们相信，“21世纪高等学校计算机规划教材——精品系列”一定能够为我国高等院校计算机教学做出应有的贡献。同时，对于工作中的欠缺和不妥之处，欢迎老师和读者提出宝贵的意见和建议。

前言

C#语言是一种完全面向对象的基于.NET的编程语言,已先后被欧洲计算机制造商协会和国际标准化组织批准为高级语言开发标准(ECMA-334、ISO/IEC 23270)。随着.NET技术的普及, C#语言已成为开发基于.NET的企业级应用程序的首选语言。

本书第1版以高度的实用性和通俗易懂的讲解,受到读者的普遍欢迎。

本书在继承第1版教材特色的基础上,结合作者多年的教学经验,并特别根据近几年教学改革的实践以及对人才培养的高标准要求,对其内容做了进一步的优化、补充和完善。本书在第1版教材的基础上做了较大改进,使各章结构更加紧凑,知识点的介绍也更突出,更适合学生学习,同时增加了一些新的技术。另外,对一些初学者比较容易糊涂的地方,也作了更为详细的阐述。

本书具有以下特色。

1. 语言简洁、通俗易懂

对一些初学者比较难以理解的知识点,尽量用简洁、形象而又浅显易懂的语言来表达。比如泛型、接口、组件、特性、纹理、图形图像用法等。

2. 代码详细、注重实用

对项目开发中比较常用的技术,都用与实际应用非常接近的例子来说明,而且有详细的代码解释。这些例子都是在项目开发中初学者迫切想实现而又不知如何做的内容,使读者学习后能很快明白具体用法,能马上编写出生动实用的应用程序。

3. 循序渐进、由浅入深

章节内容的安排采用循序渐进的办法,对于后面例子中用到的知识点,在前面章节的介绍中都有相应地铺垫,使读者从最简单的知识入手,一步步走入很有“艺术性”的各种“深奥”的编程技术中。

4. 重点介绍、适当扩充

对常用的技术,本书花费了较多的篇幅来介绍,并以图文并茂的形式,将重点内容展现出来。同时,也适当地介绍了一些比较新的编程技术,如LINQ等,使读者能够根据介绍的内容,很快对所学知识举一反三、灵活运用。

各高校在教学过程中,可以根据专业课程体系和学期总学时数,选取本书的全部或部分内容讲解,建议各章学时分配如下。

54 学时				72 学时			
第1章	2学时	第9章	8学时	第1章	2学时	第9章	10学时
第2章	4学时	第10章	2学时	第2章	6学时	第10章	2学时
第3章	4学时	第11章	2学时	第3章	4学时	第11章	2学时
第4章	4学时	第12章	2学时	第4章	4学时	第12章	4学时
第5章	2学时	第13章	2学时	第5章	4学时	第13章	2学时
第6章	8学时	第14章	6学时	第6章	10学时	第14章	6学时
第7章	2学时	第15章	2学时	第7章	4学时	第15章	6学时
第8章	4学时			第8章	6学时		

对于讲授课时为 54 学时的学校,若学时紧张,也可以将第 5 章和最后一章穿插到其他章节中讲解。

对于讲授课时为 72 学时的学校,由于第 5 章属于较高级的内容,学生在基本开发技术没有掌握之前,真正理解并将这些技术应用到实际中可能有一定难度,建议在讲解完其他章节后,再系统讲解这一章的内容,这样效果可能更好。

本书由马骏担任主编,邓居英、杨湖担任副主编,马骏对全书进行了统稿、修改和定稿。参与各章编写和代码调试等工作的还有侯彦娥、王寅、付征叶、乔保军、孙清伟、姚远、张娜等。

为了配合教学需要,本书还提供配套的教学课件、全书所有例题的源程序代码以及所有习题参考解答。读者可以与人民邮电出版社联系索取,或到人民邮电出版社教学服务与资源网(<http://ptpedu.com.cn>)上下载。

由于编者水平有限,书中难免存在错误之处,敬请读者批评指正。

编 者
2009 年 2 月

附录 A				附录 B			
附录 A	第 1 章	附录 B	第 1 章	附录 A	第 1 章	附录 B	第 1 章
附录 A	第 2 章	附录 B	第 2 章	附录 A	第 2 章	附录 B	第 2 章
附录 A	第 3 章	附录 B	第 3 章	附录 A	第 3 章	附录 B	第 3 章
附录 A	第 4 章	附录 B	第 4 章	附录 A	第 4 章	附录 B	第 4 章
附录 A	第 5 章	附录 B	第 5 章	附录 A	第 5 章	附录 B	第 5 章
附录 A	第 6 章	附录 B	第 6 章	附录 A	第 6 章	附录 B	第 6 章
附录 A	第 7 章	附录 B	第 7 章	附录 A	第 7 章	附录 B	第 7 章
附录 A	第 8 章	附录 B	第 8 章	附录 A	第 8 章	附录 B	第 8 章

目 录

第 1 篇 C#程序设计基础

第 1 章 概述.....2

1.1 C#语言及其发展简介.....2

1.1.1 C#语言的特点.....2

1.1.2 C#语言的发展过程.....3

1.1.3 C#语言与其他语言的比较.....3

1.2 Visual Studio 开发环境的功能和特点.....5

1.2.1 Microsoft.NET Framework.....6

1.2.2 应用程序模板.....7

1.3 C#应用程序的一般结构.....10

1.3.1 命名空间.....10

1.3.2 Main 方法.....12

1.3.3 文件结构.....12

1.3.4 程序代码注释.....13

1.4 断点设置与程序调试.....15

1.5 C#代码编写命名建议.....16

习题.....16

第 2 章 数据类型与流程控制.....17

2.1 数据类型.....17

2.1.1 C#数据类型的分类.....17

2.1.2 整型.....18

2.1.3 浮点型.....19

2.1.4 布尔型.....19

2.1.5 字符类型与字符串类型.....19

2.2 不同数据类型之间的转换.....20

2.2.1 显式转换与隐式转换.....20

2.2.2 装箱和拆箱.....22

2.3 常量与变量.....22

2.3.1 常量.....23

2.3.2 变量.....23

2.3.3 匿名类型的变量.....24

2.4 运算符与表达式.....24

2.4.1 运算符.....25

2.4.2 表达式.....26

2.5 控制台应用程序与格式化输出.....26

2.5.1 控制台输出.....26

2.5.2 控制台输入.....27

2.5.3 字符串的格式化输出.....27

2.5.4 数字的格式化输出.....28

2.5.5 日期和时间的格式化输出.....29

2.6 C#流程控制语句.....30

2.6.1 分支语句.....30

2.6.2 循环语句.....33

2.6.3 跳转语句.....36

2.6.4 异常处理语句.....39

习题.....41

第 3 章 常用数据类型的用法.....42

3.1 字符串.....42

3.1.1 字符串的创建与表示形式.....42

3.1.2 字符串比较.....42

3.1.3 字符串查找.....43

3.1.4 求子字符串.....44

3.1.5 字符串的插入、删除与替换.....44

3.1.6 移除首尾指定的字符.....45

3.1.7 字符串的合并与拆分.....45

3.1.8 字符串中字母的大小写转换.....46

3.1.9 String 与 StringBuilder 的区别.....46

3.2 数组.....47

3.2.1 数组的声明与初始化.....47

3.2.2 数组的秩与数组长度.....48

3.2.3 一维数组.....48

3.2.4 多维数组.....48

3.2.5 交错数组.....49

3.2.6 动态改变数组大小.....50

3.2.7 数组元素的排序与查找.....51

3.2.8 数组的统计运算及数组和字符

串之间的转换	52	4.3.1 方法的定义与调用	71
3.3 枚举	53	4.3.2 方法中的参数传递	72
3.3.1 枚举的定义	53	4.3.3 方法重载	74
3.3.2 枚举的基本用法	53	4.4 属性与索引器	75
3.4 泛型	54	4.4.1 属性	75
3.4.1 泛型的定义和引用	54	4.4.2 索引器	77
3.4.2 可空类型的泛型	55	习题	79
3.5 泛型集合	56	第5章 面向对象的高级编程	80
3.5.1 哈希集合	57	5.1 类的继承与多态性	80
3.5.2 列表和排序列表	57	5.1.1 基类和扩充类	80
3.5.3 链表	58	5.1.2 多态性	82
3.5.4 字典和排序字典	59	5.1.3 抽象类	85
3.5.5 队列	60	5.1.4 密封类	86
3.5.6 堆栈	60	5.1.5 继承过程中构造函数的处理	87
3.6 日期与时间处理	60	5.2 版本控制	88
3.7 数学运算	62	5.3 接口	90
3.8 随机数	63	5.3.1 接口的声明与实现	91
习题	64	5.3.2 显式方式实现接口	92
第4章 面向对象的编程基础	65	5.4 委托与事件	93
4.1 类	65	5.4.1 委托	93
4.1.1 类的声明与成员组织	65	5.4.2 事件	95
4.1.2 构造函数	67	5.5 反射	96
4.1.3 字段和局部变量	68	5.6 序列化与反序列化	97
4.1.4 静态成员和实例成员	69	5.6.1 二进制序列化与反序列化	97
4.1.5 访问修饰符	69	5.6.2 XML 序列化与反序列化	99
4.2 结构	69	习题	101
4.3 方法	71	第2篇 Windows 应用程序设计	
第6章 窗体、控件和组件	104	6.1.6 对话框	110
6.1 窗体与对话框	104	6.2 控件共有的基本操作	114
6.1.1 窗体应用程序的启动和停止	104	6.2.1 控件常用属性和基本操作	114
6.1.2 窗体的创建、显示、隐藏和 关闭	105	6.2.2 控件的锚定和停靠	116
6.1.3 窗体常用属性	107	6.2.3 控件的常用鼠标与键盘事件	117
6.1.4 窗体位置及外观控制	108	6.3 容器类控件和常用组件	119
6.1.5 单文档窗体和多文档窗体	110	6.3.1 分组控件 (Panel、GroupBox)	119
		6.3.2 工具提示组件 (ToolTip)	120
		6.3.3 错误提示组件 (ErrorProvider)	120

6.3.4 定时组件 (Timer)	120	7.3 文件管理	158
6.4 文本操作类控件	122	7.3.1 File 类与 FileInfo 类	158
6.4.1 标签控件 (Label、LinkLabel)	122	7.3.2 文件的复制、删除与移动	158
6.4.2 文本控件 (TextBox、Masked TextBox、RichTextBox)	122	7.3.3 文件属性	159
6.5 选择操作类控件	125	7.3.4 OpenFileDialog 与 SaveFileDialog 对话框	160
6.5.1 列表控件 (ListBox、 ComboBox)	125	7.4 文件的读写	162
6.5.2 复选控件 (CheckBox、 CheckedListBox)	128	7.4.1 文件编码	162
6.5.3 单选控件 (RadioButton)	132	7.4.2 文本文件的读写	163
6.5.4 日期时间选择控件 (DateTimePicker)	133	7.4.3 二进制文件的读写	165
6.6 图像操作类控件和组件	135	7.5 FileSystemWatcher 组件	166
6.6.1 图像与动画控件 (PictureBox)	135	习题	167
6.6.2 图像列表组件 (ImageList)	136	第 8 章 图形图像处理	168
6.7 菜单、工具栏与状态栏	137	8.1 图形图像绘制基础知识	168
6.7.1 菜单控件 (MenuStrip)	137	8.1.1 GDI+概述	168
6.7.2 快捷菜单控件 (Context MenuStrip)	139	8.1.2 Graphics 类	170
6.7.3 工具栏控件 (ToolStrip)	139	8.1.3 颜色	170
6.7.4 状态栏控件 (StatusStrip)	139	8.1.4 Paint 事件	171
6.8 视图操作类控件	142	8.2 绘制基本图形	172
6.8.1 列表视图控件 (ListView)	142	8.2.1 创建画笔	172
6.8.2 树形视图控件 (TreeView)	144	8.2.2 绘制直线	172
6.9 其他常用控件	147	8.2.3 绘制矩形	177
6.9.1 选项卡控件 (TabControl)	147	8.2.4 绘制多边形	177
6.9.2 面板复合控件 (SplitContainer)	149	8.2.5 绘制曲线	179
6.9.3 任务栏图标组件 (NotifyIcon)	150	8.2.6 绘制椭圆和扇形	180
习题	152	8.3 填充图形	181
第 7 章 目录与文件管理	153	8.3.1 创建画刷	181
7.1 系统环境相关类	153	8.3.2 填充单色	182
7.1.1 Environment 类	153	8.3.3 填充渐变色	183
7.1.2 DriveInfo 类	154	8.3.4 填充阴影	184
7.2 目录管理	154	8.3.5 填充纹理	185
7.2.1 Directory 类与 DirectoryInfo 类	155	8.3.6 填充路径	186
7.2.2 Path 类	156	8.4 图像处理	187
7.2.3 目录的创建、删除和移动	156	8.4.1 绘制图像	187
7.2.4 FolderBrowserDialog 对话框	157	8.4.2 保存图像	188
		8.4.3 图像的拉伸与反转变换	191
		8.5 图形与图像的平移、旋转和缩放	193
		8.6 文字处理	194
		习题	195

第9章 ADO.NET 与数据操作196

9.1 ADO.NET 简介	197
9.1.1 数据访问技术及其发展概述	197
9.1.2 ADO.NET 数据访问模型	197
9.2 数据库与数据访问工具	198
9.2.1 SQL Server 数据库分类	198
9.2.2 数据访问可视化工具	200
9.3 ADO.NET 数据访问对象	203
9.3.1 SqlConnection 对象	204
9.3.2 SqlCommand 对象	205
9.3.3 SqlDataAdapter 对象	209
9.3.4 DataTable 对象	210
9.3.5 DataSet 对象	212
9.4 数据绑定技术	213
9.4.1 简单数据绑定与复杂数据绑定	213
9.4.2 数据源组件 (BindingSource)	215
9.4.3 导航控件 (BindingNavigator)	217
9.5 DataGridView 控件	220
9.5.1 默认功能	220
9.5.2 绑定数据源	221
9.5.3 标题及行列控制	224
9.5.4 单元格控制	227
9.5.5 异常处理	230
9.6 数据处理	233
9.6.1 图像数据处理	233
9.6.2 关联表数据处理	236
9.7 存储过程	240
9.8 语言集成查询 (LINQ)	244
9.8.1 LINQ 简介	244
9.8.2 查询表达式	246
9.8.3 对象关系设计器 (O/R 设计器)	249
习题	251

第10章 报表设计252

10.1 水晶报表基础知识	252
10.1.1 水晶报表的分类	252
10.1.2 嵌入式水晶报表设计器	253
10.1.3 报表节	253
10.1.4 报表数据源	254
10.2 水晶报表的设计与显示	254
10.2.1 水晶报表对象模型	254
10.2.2 显示与打印水晶报表	256
习题	264

第11章 类库与控件库设计265

11.1 特性 (Attribute)	265
11.2 类库设计	266
11.2.1 设计类库	267
11.2.2 调用类库	268
11.3 用户控件	269
11.3.1 在属性窗口中显示属性提示	269
11.3.2 制作按钮用户控件	271
11.4 控件库设计	275
11.4.1 设计控件库	275
11.4.2 调用控件库	276
习题	276

第12章 正则表达式277

12.1 正则表达式及其相关类	277
12.1.1 正则表达式简介	277
12.1.2 正则表达式相关类	280
12.2 Web 信息浏览与搜索	285
12.2.1 WebBrowser 控件	285
12.2.2 利用正则表达式搜索 Web 资源	287
习题	291

第3篇 Web 应用程序设计**第13章 Web 应用程序设计****基础**294

13.1 Web 应用程序开发工具	294
13.2 页面设计基础	295

13.2.1 HTML	295
13.2.2 CSS	297
13.2.3 JavaScript	299
13.3 Web 应用程序中的常用对象	300
13.3.1 Response 与 Request	301

13.3.2 Application 与 Session 对象	301	14.4 导航控件	327
13.4 网页切换与网页间的数据传递	301	14.4.1 SiteMapPath 控件	327
13.4.1 网页的切换	302	14.4.2 Menu 控件	329
13.4.2 网页间的数据传递	302	14.4.3 TreeView 控件	331
习题	302	习题	334
第 14 章 ASP.NET Web 服务器		第 15 章 Web 应用程序开发	
控件	303	实例	335
14.1 标准控件	303	15.1 系统分析与总体规划	335
14.1.1 简单控件	304	15.1.1 需求分析	335
14.1.2 Table 控件	306	15.1.2 技术处理	336
14.1.3 HiddenField 控件	309	15.1.3 总体规划	336
14.2 数据操作控件	310	15.2 系统架构设计	336
14.2.1 SqlDataSource 组件	310	15.2.1 系统功能结构	337
14.2.2 GridView 控件	313	15.2.2 文件组织	337
14.2.3 DataList 控件	315	15.2.3 数据库结构	338
14.2.4 DetailsView 控件	319	15.3 前台功能模块设计	338
14.2.5 FormView 控件	320	15.3.1 首页设计	339
14.3 验证控件	320	15.3.2 单记录多选类测评表页面设计	342
14.3.1 ValidationSummary 控件	321	15.3.3 多记录多选类测评表页面设计	343
14.3.2 RequiredFieldValidator 控件	321	15.3.4 数据编辑类测评表页面设计	346
14.3.3 RangeValidator 控件	322	15.4 后台功能模块设计	347
14.3.4 CompareValidator 控件	323	15.4.1 管理员登录页面设计	347
14.3.5 RegularExpressionValidator		15.4.2 系统管理页面设计	347
控件	325	15.4.3 统计汇总表设计	348
14.3.6 CustomValidator 控件	326	15.4.4 报表打印与输出设计	348

第1篇

C#程序设计基础

C#语言是在C和C++的基础上重新构造的、语法与C++和Java都比较相似的、基于.NET框架支持的一种完全面向对象的、类型安全的编程语言，也是.NET的首选编程语言。微软公司设计C#语言的目的主要是为了简化网络应用编程的难度。从开发效率来讲，C#为应用程序开发人员提供了快速的开发手段，但又保持了C++语言的特点和优点。从继承性来讲，C#在更高层次上重新实现了C和C++。从语法形式和易用性来讲，C#语言几乎综合了目前流行的所有高级语言的优点，提供了一种语法优雅、功能完善而又容易使用的外在表现形式。

本篇主要介绍C#语言程序设计基础知识。

为了让读者把注意力集中在对基本知识的理解上，在介绍其他类型的应用程序之前的这些章节中，均使用控制台应用程序作为例子，但是这些语言本身的基本用法同样适用于Windows窗体应用程序和ASP.NET Web应用程序。

控制台应用程序使用标准命令行输入和输出，而不是在图形窗体上显示。控制台应用程序使用System命名空间中的类处理输入和输出。

C#语言的基本语法和面向对象的设计思想是实际应用项目开发的基础，希望读者能很好地理解和掌握。

第 1 章 概述

C#语言和 Microsoft.NET 框架简化了软件开发的复杂度,利用 C#语言和基于.NET 框架的 Visual Studio 2008 (简称 VS2008)集成开发平台,程序员可以非常方便地开发出各种应用程序。

这一章我们对 C#语言、.NET 框架以及 VS2008 作一个基本的了解。

1.1 C#语言及其发展简介

C# (读作“see sharp”)的叫法很有创意,意思是让我们看看这种语言多么锋利无比。C++不是有两个“+”号吗,那就让它有 4 个“+”号吧,这就是“#”的来历。

1.1.1 C#语言的特点

C#语言具有以下主要特点。

(1) 简洁的语法

C#语言和 Java 语言一样,使用了统一的操作符,淘汰了 C++语言中乱糟糟的表示符号和伪关键字,使用最简单、最常见的形式进行描述。

(2) 精心的面向对象设计

C#语言是完全按照面向对象的思想来设计的,因此,它具有面向对象所应有的一切特性,如封装、继承、多态性等。

在类的继承方面,C#语言只允许单继承,即一个类不会有多个基类,从而避免了类型定义的混乱。

在 C#语言中,每种类型都是一个对象,不存在全局函数、全局变量等概念,所有常量、变量、属性、方法、索引、事件等都必须封装在类中,从而使代码具有更好的可读性,也避免了发生命名冲突的可能。

(3) 与 Web 的紧密结合

在 C#语言中,对于复杂的 Web 编程和其他网络编程看起来更像是对本地对象进行操作,从而简化了大规模、深层次的分布式开发。用 C#语言构建的 Web 组件能够方便地作为 Web 服务(Web Service)并可以通过 Internet 被运行在任何操作系统上的任何语言所调用。

(4) 可靠的安全性与错误处理

语言的安全性与错误处理能力是衡量一种语言是否优秀的重要依据。C#语言可以消除许多软件开发中的常见错误,并提供了包括类型安全在内的完整的安全性能。

在默认情况下,从 Internet 和 Intranet 下载的代码都不允许访问任何本地文件和资源;C#语言不允许使用未初始化的局部变量,并提供了完善的边界检查与溢出检查等功能。内存管理中的垃圾回收机制也极大地减轻了开发人员对内存管理的负担。

(5) 版本处理技术

C#语言内置了版本控制功能,如对方法重载和接口的处理方式以及对特性 (Attribute) 的支持等,从而保证方便地开发和升级复杂的软件。

(6) 灵活性和兼容性

在托管状态下,C#语言不能使用指针,而是用委托 (Delegate) 来模拟指针的功能。如果确实需要在类或者类的方法中使用指针,只需要声明这些代码为非托管的代码就可以了。另外,虽然 C#语言不支持类的多继承,但是却可以通过接口来实现多继承。

兼容性是指 C#语言允许与 Win32 API (Application Programming Interface) 进行交互操作,允许 C#语言组件与其他语言组件间的交互操作等。

1.1.2 C#语言的发展过程

C#语言的发展过程主要经历了以下阶段。

2000 年,C#语言诞生。

2003 年,微软公司发布了 C#语言规范 1.2 (简称 C#1.2),VS.NET 2003 使用的是 C#1.2。

2005 年,微软公司发布了 C#语言规范 2.0 (简称 C#2.0),VS2005 使用的是 C#2.0。

2007 年,微软公司发布了 C#语言规范 3.0 (简称 C#3.0),VS2008 使用的是 C#3.0。

由于 C#还在快速发展,因此版本更新较快。C#2.0 在 C#1.2 的基础上,增加了泛型、迭代、分部类、匿名方法等,C#2.0 版本已经被国际标准化组织定为高级语言开发标准。

C#3.0 在 C#2.0 的基础上,又增加了查询表达式、Lambda 表达式、自动实现的属性、匿名变量、隐式类型的本地变量和数组,以及扩展方法等功能,C#3.0 由于刚推出不久,尚未通过标准化组织审核。

1.1.3 C#语言与其他语言的比较

本小节将 C#语言与目前常用的高级语言做一个简单的比较,以便熟悉这些语言的读者可以更好地了解 C#语言的特性,更快地掌握 C#语言的编程方法。

1. C#语言与 C++语言的比较

C#语言对 C++语言进行了多处改进,现将主要区别介绍如下。

- 编译目标:C++代码直接编译为本地可执行代码,而 C#语言默认编译为中间语言 (IL) 代码,执行时再通过 JIT (Just-In-Time) 编译器将需要的模块临时编译成本地代码。

- 内存管理:C++语言要求程序员显式地删除动态分配给堆的内存,而 C#语言不需要这么做,C#语言采用垃圾回收机制自动在合适的时机回收不再使用的内存。

- 指针:C++语言中大量地使用指针,用起来比较复杂,对不熟练的程序员来说,编写的程序容易引起内存泄露以及管理漏洞;而 C#语言使用对类实例的引用,虽然在非类型安全的状态下 C#也可以使用指针,不过,一般情况下 C#程序员没必要这样做。

- 字符串处理:在 C#语言中,字符串是作为一种基本数据类型来对待的,因此 C#语言比 C++语言中对字符串的处理要简单得多。

- C++语言允许类的多继承,而 C#语言只允许类的单继承,但可以通过接口实现多继承。

在后面的学习中我们会发现,C#语言与 C++语言相比还有很多不同和改进之处,包括一些细节上的差别,这里就不一一列举了。

2. C#语言与 Java 语言的比较

从语法上讲,C#语言和 Java 语言非常相似,只是在细节上有一些差别。实际上,C#语言和 Java 语言的主要差别不是在语言本身,而是在内部功能实现上以及性能上不一样。

Java 程序需要一个运行环境 JRE(Java Runtime Environment)来执行代码,但 JRE 只限于在 Java 这一门语言中使用。C#语言也需要一个运行环境 CLR(Common Language Runtime),但是 CLR 提供了对支持 .NET 框架的所有语言的支持。

Java 源代码可以被编译成字节代码的一种中间状态,然后由已提供的虚拟机来执行这些字节代码。C#代码也被编译成一种中间状态,称为中间语言(IL),但是 IL 代码则被传输到由 CLR 管理的执行进程上,然后通过 CLR 的 JIT 编译器编译成本地代码执行。

C#语言与 Java 语言相比也有很多不同和改进之处。例如,C#语言的文件名不受文件中类名的限制,而在 Java 语言中则有此限制。另外,C#语言也提供了一些在 Java 语言中没有的功能,如运算符重载、装箱和拆箱、结构以及方法隐藏等。

表 1-1 所示为 C#语言和 Java 语言的比较。

表 1-1

C#语言和 Java 语言的比较

项 目	C#	Java
运行环境	可在具有 CLR 的平台上运行	可在具有 JVM 的平台上运行
完全面向对象	是	是
多重继承	不支持	不支持
内存管理	使用垃圾回收机制管理内存	使用垃圾回收机制管理内存
异常处理	try-catch-finally	try-catch-finally
指针	只能在非托管的代码段内使用	不支持
类型安全性验证	强制类型验证	强制类型验证
命名空间	支持	支持
布尔值	只能为 true 或者 false	只能为 true 或者 false
变量初始化	不能使用未初始化的变量	不能使用未初始化的变量
中间语言处理	将 MSIL 转换为 JIT 机器码	Java 字节码
访问修饰符	public、protected、private、internal、partial	public、protected、private、friendly
Web Service	支持	支持

3. C#语言与 VB.NET 语言的比较

和 C#语言一样,VB.NET 语言也是基于 .NET Framework 和 CLR 的高级语言。但是 C#语言有一些 VB.NET 语言所不具备的独有的特性,如 C#语言可以使用非托管代码、移位操作符、内嵌的文档(XML)和运算符重载等。在发展前景上,由于 C#语言一开始就是完全按照面向对象的思想来设计的,而且它使用的全部是 .NET 框架定义的语法格式,不存在考虑与 .NET 之前版本兼容的问题,因此给人的感觉是结构清晰,语法简洁、优雅。另外,C#语言作为一种高级语言标准,其基本的内部实现形式是公开的,因此更容易被多种平台(例如 UNIX、Linux 等)接受和广泛应用。当然,VB.NET 语言也有其自身的优点,如以前学习过 VB 的人既可以使用 .NET 之前的语法格式及函数,也可以使用 .NET 规定的语法格式和面向对象的设计思想。

1.2 Visual Studio 开发环境的功能和特点

基于.NET的Microsoft Visual Studio是.NET平台下非常强大的软件开发工具,无论是软件服务商,还是企业级应用程序的部署与发布,Visual Studio开发平台都可以提供近乎完美的解决方案。

在Visual Studio开发环境下,提供了一整套的软件开发和测试工具,包括设计、编码、编译调试、与数据库的互操作等基本功能和基于开放架构的服务器组件开发平台、企业开发工具和应用程序重新发布工具以及性能评测报告等高级功能。

表1-2所示为基于.NET的Visual Studio系列开发平台提供的编程工具及特点。

表1-2 基于.NET的Visual Studio系列开发平台提供的编程工具及特点

功 能	描 述
Windows 窗体设计器	提供图形化设计界面,通过对窗体中的控件进行拖放,可以快速创建应用程序的用户界面
Windows 窗体工具	提供了Windows窗体设计器、Windows应用程序模板、基本的项目引用和初始代码,以帮助用户创建标准Windows窗体应用程序
Web 窗体工具	提供了Web窗体设计器、ASP.NET Web应用程序模板、基本的项目引用和初始代码,以帮助用户创建以浏览器作为主界面的Web窗体应用程序
XML Web Services 工具	提供了一个ASP.NET Web Service模板,可以用来构成Web应用程序的基本结构。此时,Web应用程序的基本架构将构建在Web服务器和本地解决方案文件之上
多语言支持	开发环境集成了所有的.NET平台编程语言,包括Visual C#语言
数据访问	包括用于创建数据库应用程序的组件、可视化数据库工具以及一个可靠的ADO.NET类集,使开发人员可以方便地操作所有类型的数据
错误处理	包括支持跨语言调试的调试工具以及结构化异常类
向导	可以用来快速创建复杂的通用任务

基于.NET的Visual Studio开发平台主要经历了以下阶段。

2003年,微软公司发布了Visual Studio.NET 2003(简称VS.NET 2003)和.NET Framework 1.1。

2005年,微软公司发布了Visual Studio 2005(简称VS2005)和.NET Framework 2.0。

2006年,微软公司发布了.NET Framework 3.0。

2007年,微软公司发布了Visual Studio 2008(简称VS2008)以及.NET Framework 3.5。

2008年,微软公司发布了Visual Studio 2008 SP1以及.NET Framework 3.5 SP1。

对Visual Studio的每一个版本来说,与它的前一个版本相比,都有非常多的改进。下面列出Visual Studio2008与Visual Studio2005相比,我们感兴趣的几个方面。

- Visual Studio2005不支持SQL Server 2008,而Visual Studio2008 SP1开始全面支持SQL Server 2008。

- Visual Studio2008在Web应用程序开发方面与Visual Studio2005相比变化很大,如新的设计视图和CSS设计工具、JavaScript的单步调试和智能提示、内置的AJAX等。

- Visual Studio2008在Windows应用程序开发方面提供了更多的功能,如专为P2P提供的类、支持XNA3.0、智能客户端发布更简单等。

Visual Studio2008主要有以下版本。

(1) Visual Studio 2008 速成版 (Express Edition): 该版本为免费版本, 功能有限, 适合初学者学习使用。

(2) Visual Studio 2008 标准版 (Standard Edition): 标准版适合一般软件公司。

(3) Visual Studio 2008 专业版 (Professional Edition): 该版本功能完善, 适合专业软件公司。

(4) Visual Studio 2008 团队版 (Team Edition): 团队版在 Visual Studio 2008 Professional Edition 的基础上提供了更高级的开发工具。

除了上面几种版本外, 还有团队套装版 (Team Suite)、团队开发服务器版等。

本书的所有例子均在安装了 SP1 的 Visual Studio 2008 Professional Edition 开发环境下调试通过。

1.2.1 Microsoft.NET Framework

Microsoft.NET Framework (简称.NET 框架) 是生成、运行.NET 应用程序和 Web Service 的组件库, 它包括两个主要组件, 一个是公共语言运行库 (简称运行库), 另一个是类库。运行库提供.NET 应用程序所需要的核心服务, 类库为开发和运行.NET 应用程序提供了各种支持。

图 1-1 所示为公共语言运行库和类库与应用程序及其他系统之间的关系。

.NET 框架实现的目标如下。

- 提供一个对各种语言都一致的面向对象的编程环境。

- 提供一个将软件部署和版本控制冲突最小化的代码执行环境。

- 提供一个可提高代码执行安全性的代码执行环境 (包括托管代码和非托管代码)。

- 提供一个可消除脚本环境或解释环境的性能问题的代码执行环境。

- 使开发人员的开发经验在 Windows 应用程序和 Web 应用程序中保持一致。

- 按照工业标准生成所有通信, 以确保基于.NET 框架的代码可与任何其他代码集成。

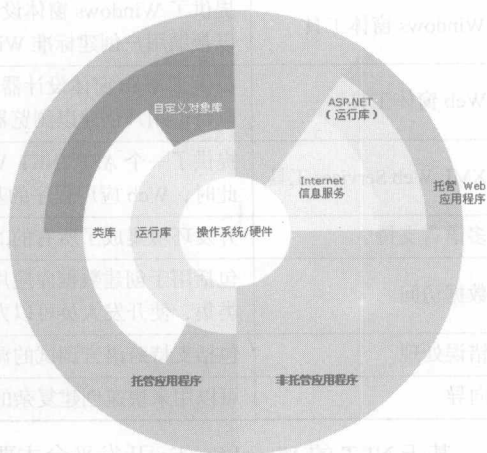


图 1-1 .NET Framework 环境

用基于.NET 框架开发的应用程序, 不论使用的是哪种高级语言, 均必须在安装了.NET 框架的计算机上才能运行。这种架构与 Java 应用程序必须由 Java 虚拟机支持相似。

目前, 除了基于.NET 框架的 C#语言、VB.NET 语言、C++.NET 语言以及和 Java 语言语法完全相同的 J#语言外, 还有基于.NET 框架的 FORTRAN 语言、Pascal 语言、COBOL 语言、PERL 语言、Python 语言、Eiffel 语言等其他高级语言。

1. 运行库

运行库是.NET 框架的基础, 可被看作是一个在执行时管理代码的代理, 提供诸如内存管理、线程管理、远程处理等核心服务, 而且还强制实施严格的类型安全以及确保安全性和可靠性的其他形式的代码的准确性。

以运行库为目标的代码称为托管代码, 不以运行库为目标的代码称为非托管代码。

使用.NET 框架提供的编译器可以直接将源程序编译为.EXE 或者.DLL 文件, 但是需要注意的是, 此时编译出来的程序代码并不是 CPU 能直接执行的机器代码, 而是一种中间语言 IL (Intermediate Language) 代码。