



邱仲潘 宋智军 编著

Visual Basic 2010

中文版

从入门到精通



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

Visual Basic 2010中文版

从入门到精通

邱仲潘 宋智军 编著

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

本书坚持以语言为基础和应用为主导的编写原则,介绍如何使用Visual Basic 2010开发Windows窗体应用程序和ASP.NET应用程序。为了更好地帮助读者在短时间内掌握使用Visual Basic语言开发各种应用程序的知识点和编程技巧,全书的基础知识介绍清晰,理论联系实际,具有很强的操作性。本书还提供了大量的通过测试可运行的完整实例,这些实例都有设计步骤、代码详解和程序运行结果等,适合初学和中级编程人员。

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

Visual Basic 2010中文版从入门到精通/邱仲潘,宋智军编著.—北京:电子工业出版社,2011.1
ISBN 978-7-121-12502-7

I. ①V… II. ①邱… ②宋… III. ①BASIC语言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字(2010)第240204号

责任编辑:李红玉

文字编辑:徐磊

印刷:北京天竺颖华印刷厂

装订:三河市鑫金马印装有限公司

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

北京市海淀区翠微东里甲2号 邮编:100036

开本:787×1092 1/16 印张:18 字数:460千字

印次:2011年1月第1次印刷

定 价:37.00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

在Visual Studio 6.0之后，微软发布了.NET平台，其后的Visual Studio都是基于.NET平台的。无论是开发环境的改善，还是新的开发语言C#的引入，都没有使得Visual Studio能够再创辉煌，再续经典。每次新版本的发布，程序员们只看到安装程序越来越大，整个开发环境越来越臃肿，却没有看到多少革命性的变化，这多少有点让我们这些天天使用Visual Studio的程序员们有些失望，难道真的是“英雄迟暮”？

随着Visual Studio 2010的发布，微软同时宣称会将它打造成跟Visual Studio 6.0一样的经典产品，这又让我们就快冰凉的心重燃希望。新版本什么样？有什么新功能？真的会成为下一个经典吗？关心Visual Studio的人都在问，迫切地想先睹为快。现在机会来了，新版本的发布，让我们有机会一睹VS2010的芳容，看看微软将如何一步步打造这个未知的“经典”产品。

根据微软发布的一份官方文档宣称，Visual Studio 2010和.NET Framework 4.0将在以下5个方面有所创新。

◆民主化的应用程序生命周期管理

在一个组织中，应用程序生命周期管理（ALM）将牵涉到多个角色。但是在传统意义上，这一过程中的每个角色并不是完全平等的。Visual Studio Team System 2010将坚持打造一个功能平等、共同分担的平台，以用于组织内的应用程序生命周期的管理过程。

◆顺应新的技术潮流

每年，业界内的新技术和新趋势层出不穷。通过Visual Studio 2010，微软将为开发者提供合适的工具和框架，以支持软件开发中最新的架构、开发和部署。

◆让开发商惊喜

从Visual Studio的第一个版本开始，微软就将提高开发人员的工作效率和灵活性作为自己的目标。Visual Studio 2010将继续关注并且显著地改进开发者最核心的开发体验。

◆下一代平台浪潮的弄潮儿

微软将继续投资于市场领先的操作系统、工具软件和服务器平台，为客户创造更高的价值。使用Visual Studio 2010，将可以在新一代的应用平台上，为你的客户创造令人惊奇的解决方案。

◆跨部门的应用

客户将在不同规模的组织内创建应用，跨度从单个部门到整个企业。Visual Studio 2010将确保在这么宽泛的范围内的应用开发都得到支持。

看了这么高度的概括，相信大家都会迷迷糊糊，不知所云。这就是大公司“枪文”的厉害之处：让你听得云里雾里，但是又让你觉得它的东西很好。实际上，上面这段官方文

档可以翻译成程序员更好理解的内容。

“Visual Studio 2010将致力于引领下一代平台技术的发展、提高开发人员的工作效率和热情、创建突破性的应用程序、推动云计算等新兴技术趋势、合理化应用程序生命周期管理（ALM）。另外，Visual Studio 2010将支持统一建模语言（UML）和特定域语言（DSL），它将可以为各类开发人员提供合适的工具，而这也是微软更大模型平台的一部分。”

这样的介绍是不是更好理解一些呢？上面的介绍都是高度地概括VS2010的新特性，实际上，我们希望能够看到一些更加具体、更加实际的内容。所以，在接下来的内容中，会以Visual Studio 2010为基础，带领大家一同来看看Visual Studio 2010中的新东西，从现在开始体验下一代Visual Studio。根据微软提供的一份指导文档为蓝本，将依次介绍其全新的基于WPF创建的IDE、增强的代码编辑器、Visual Basic中所带来的一些令人激动不已的新特性，等等。

关于本书

本书坚持以语言为基础和应用为主导的编写原则，介绍如何使用Visual Basic 2010开发Windows窗体应用程序和ASP.NET应用程序。为了更好地帮助读者在短时间内掌握使用Visual Basic语言开发各种应用程序的知识点和编程技巧，全书的基础知识介绍清晰，理论联系实际，具有很强的操作性。本书还提供了大量的通过测试可运行的完整实例，这些实例都有设计步骤、代码详解、程序运行结果等，不但复习了前面所学的内容，而且还增加了一定量的创作技巧。对于容易出现问题的地方，则以“注”的方式介绍常用的技巧和注意事项。

本书由邱仲潘、宋智军编著，同时感谢冯姝慧、邹文、邓欣欣、王帅、朱敏、张朋丽等给予的帮助和支持。同时非常感谢电子工业出版社的编辑和出版，他们为本书的选题策划、编辑加工和出版发行付出了辛勤的劳动。由于编者水平有限，加之时间仓促，书中难免有疏漏和不足之处，恳请专家和广大读者指正。

为方便读者阅读，若需要本书配套资料，请登录“北京美迪亚电子信息有限公司”（<http://www.medias.com.cn>），在“资料下载”页面进行下载。

目 录

第1章 Visual Basic 2010概述	1	1.5.5 运行程序	17
1.1 Visual Basic简介	1	1.5.6 保存项目	17
1.1.1 Visual Basic的诞生及发展	1	第2章 Visual Basic 2010编程基础	19
1.1.2 .NET Framework 4概述	2	2.1 编码基础	19
1.1.3 Visual Basic基本概念及常用术语	3	2.1.1 关键字和标识符	19
1.2 Visual Basic 2010中的新增功能	4	2.1.2 编码规则	19
1.2.1 自动实现的属性	4	2.1.3 注释规则	20
1.2.2 集合初始值设定项	5	2.2 数据类型	20
1.2.3 隐式行断续	5	2.2.1 数值数据类型	22
1.2.4 Lambda表达式	6	2.2.2 字符及字符串类型	22
1.2.5 等价类型支持	7	2.2.3 布尔类型	23
1.2.6 共变和逆变	7	2.2.4 日期时间类型	23
1.2.7 定位到	7	2.2.5 对象类型	23
1.2.8 突出显示引用	8	2.2.6 用户自定义的类型	23
1.2.9 使用时生成	9	2.3 常量	24
1.2.10 IntelliSense建议模式	9	2.4 变量	24
1.3 Visual Studio 2010的运行环境及安装	9	2.4.1 声明变量	24
1.3.1 安装Visual Studio 2010	10	2.4.2 给变量赋值	25
1.3.2 选择默认环境设置	11	2.4.3 作用域	25
1.4 认识Visual Studio 2010集成开发环境	12	2.4.4 静态变量	26
1.4.1 菜单栏与工具栏	12	2.4.5 类型转换	27
1.4.2 解决方案资源管理器	13	2.5 运算符和表达式	28
1.4.3 工具箱和属性窗口	13	2.5.1 算术运算符	28
1.4.4 设计器窗口	14	2.5.2 关系运算符	29
1.4.5 代码编辑器	14	2.5.3 逻辑运算符	30
1.4.6 对象浏览器	15	2.5.4 字符运算符	30
1.5 Visual Basic 2010应用程序	15	2.5.5 运算符的优先级	30
1.5.1 创建Visual Basic 2010应用程序	16	2.5.6 结合性	31
1.5.2 添加控件	16	2.5.7 表达式	32
1.5.3 设置属性	16	2.6 常用函数	32
1.5.4 添加代码	16	2.6.1 数学函数	32
		2.6.2 字符处理函数	33
		2.6.3 随机数函数	33
		2.6.4 转换函数	34

2.6.5 日期函数	34	4.5.1 数组的引用	65
第3章 流程控制语句	35	4.5.2 数组元素的输入	65
3.1 程序的基本结构	35	4.5.3 数组元素的输出	66
3.2 顺序结构	35	4.5.4 数组元素的复制	67
3.2.1 赋值语句	35	4.6 数组元素的特殊操作	68
3.2.2 复合赋值语句	36	4.6.1 使用Erase语句对数组 重新初始化	68
3.2.3 输入数据	36	4.6.2 使用For Each...Next 循环访问数组	69
3.2.4 数据的输出	38	4.6.3 使用Lbound和Ubound 函数取数组上下界	69
3.3 分支结构	40	第5章 过程和函数	71
3.3.1 If语句	40	5.1 过程的概念	71
3.3.2 情况语句 (Select Case语句)	46	5.2 过程的分类	71
3.4 循环结构	48	5.2.1 按用途分类	71
3.4.1 For...Next循环语句	48	5.2.2 按调用方式分类	72
3.4.2 While循环语句	49	5.3 Sub过程	72
3.4.3 Do循环语句	50	5.3.1 通用过程	72
3.4.4 枚举循环语句	53	5.3.2 事件过程	73
3.4.5 循环嵌套语句	53	5.3.3 调用Sub过程	74
3.5 其他流程控制语句	54	5.4 Function过程	75
3.5.1 GoTo语句	54	5.4.1 Function过程的定义	75
3.5.2 With语句	55	5.4.2 Function过程的调用	76
3.5.3 Stop语句和End语句	55	5.4.3 调用其他模块中的过程	76
3.5.4 Exit语句	56	5.5 过程的作用域	77
3.5.5 DoEvents语句	56	5.6 参数传递	77
第4章 数组	58	5.6.1 形参和实参	77
4.1 数组概述	58	5.6.2 按值传递	78
4.2 声明数组	58	5.6.3 按地址传递	79
4.2.1 声明一维数组	58	5.6.4 数组作为传递参数	80
4.2.2 声明多维数组	59	5.6.5 对象参数	82
4.2.3 声明交错数组	60	5.6.6 可选参数与可变参数	82
4.3 数组的初始化	60	5.7 递归调用	85
4.3.1 一维数组的初始化	61	第6章 面向对象编程基础	88
4.3.2 多维数组的初始化	62	6.1 类	88
4.3.3 交错数组的初始化	62	6.1.1 类的定义	88
4.4 动态数组	63	6.1.2 类的字段和属性	90
4.4.1 声明动态数组	63	6.1.3 类与名空间	91
4.4.2 保留动态数组的内容	64		
4.5 数组的基本操作	65		

6.1.4 类的方法	92	7.3.4 组合类控件	127
6.1.5 类的事件	93	7.3.5 其他常用控件	129
6.1.6 类的构造函数	94	7.4 窗体和控件中的名称	130
6.2 对象	95	第8章 文件系统	131
6.2.1 创建对象	95	8.1 Visual Basic 2010提供的3种 文件I/O途径	131
6.2.2 对象之间的关系	96	8.2 传统文件方法概述	131
6.2.3 对象成员	96	8.3 文件系统对象	132
6.2.4 将对象传递给过程	97	8.3.1 FileSystemObject对象	132
6.3 继承和派生	98	8.3.2 Driver对象	134
6.3.1 继承的基础知识	98	8.3.3 Folder对象	135
6.3.2 重写派生类的属性和方法	98	8.3.4 File对象	137
6.3.3 创建派生类	99	8.3.5 TextStream对象	138
6.4 重载	100	8.4 System.IO模型	139
6.5 接口	100	8.4.1 File类	140
6.5.1 接口定义	101	8.4.2 FileStream类	141
6.5.2 创建和实现接口	101	8.4.3 StreamReader和 StreamWriter类	142
6.5.3 何时使用接口	102	8.4.4 BinaryReader和 BinaryWriter类	144
6.5.4 接口实例	103	8.4.5 Directory类	145
6.6 封装	105	8.5 读取文件	148
6.7 多态	106	8.6 写入文件	148
6.7.1 基于继承的多态性	106	第9章 多媒体编程技术	150
6.7.2 基于接口的多态性	107	9.1 多媒体环境	150
6.8 委托	108	9.1.1 多媒体服务描述	150
6.8.1 委托和事件	108	9.1.2 MCI设备	150
6.8.2 声明使用现有委托类型 的事件	108	9.1.3 多媒体文件类型	151
6.8.3 委托变量和参数	109	9.2 媒体API	151
第7章 窗体和常用控件	110	9.3 媒体播放模式	152
7.1 窗体	110	9.3.1 独立模式	152
7.1.1 创建窗体	110	9.3.2 时钟模式	152
7.1.2 窗体的属性	112	9.4 使用多媒体控件编程	152
7.1.3 窗体的方法	114	9.4.1 MediaPlayer控件	152
7.1.4 窗体的事件	115	9.4.2 Multimedia MCI控件	158
7.2 窗体设计器	116	第10章 图形图像处理	162
7.3 控件	116	10.1 图形图像基础	162
7.3.1 可调节类控件	117		
7.3.2 选择类控件	120		
7.3.3 显示类控件	121		

10.1.1 坐标系统	162	11.2.2 数据定义语言	190
10.1.2 图形控件	164	11.2.3 数据操作语言	190
10.1.3 PictureBox控件和Image控件 的异同点	164	11.2.4 数据查询语言	191
10.1.4 图形方法	165	11.2.5 数据控制语言	191
10.1.5 绘图属性	165	11.3 使用DAO	191
10.1.6 图形显示方法	165	11.3.1 DAO简介	192
10.2 GDI+	166	11.3.2 DAO的使用	192
10.2.1 GDI+概述	166	11.4 使用ADO.NET	196
10.2.2 GDI+方法	166	11.4.1 ADO.NET 4简介及 新增功能	197
10.2.3 GDI+主要命名空间	167	11.4.2 DataSet	198
10.2.4 GDI+的坐标系统	167	11.4.3 ADO.NET的使用	199
10.3 用GDI+绘图	171	11.5 创建数据库	201
10.3.1 创建Graphics对象	171	11.5.1 向数据库中添加表	201
10.3.2 从Image对象创建	172	11.5.2 添加键	202
10.4 绘制和操作图形	172	11.5.3 添加数据	202
10.4.1 创建Graphics对象	172	11.6 实现数据绑定	203
10.4.2 用Pen绘制图形	172	第12章 网络编程	205
10.4.3 用Brush填充图形	173	12.1 VBScript	205
10.5 颜色控制	177	12.1.1 Script简介	205
10.5.1 用不透明和半透明的 直线绘制图形	178	12.1.2 VBScript编程	205
10.5.2 用合成模式控制通道混合	178	12.2 网络编程基础	207
10.6 绘制直线、矩形和椭圆	180	12.2.1 网络传输协议	208
10.7 绘制弧线	181	12.2.2 使用套接字编程	209
10.8 绘制多边形和折线	182	12.3 ASP.NET	215
10.9 绘制曲线	183	12.3.1 ASP.NET 4简介及 新增功能	216
10.10 图像处理	184	12.3.2 配置IIS管理器	217
10.10.1 导入图像	184	12.3.3 创建和配置Web应用程序	218
10.10.2 平铺图像	185	12.3.4 事件处理	224
10.10.3 调整和缩放图像	185	12.3.5 输入验证	225
10.10.4 翻转和旋转图像	185	12.3.6 状态管理	231
第11章 数据库编程	187	12.4 Web服务	237
11.1 数据库基础	187	12.4.1 Web服务概述	237
11.1.1 数据库设计	187	12.4.2 使用Web服务的好处	238
11.1.2 使用Access创建数据库	188	12.4.3 Web服务体系结构	238
11.2 SQL语言	189	12.4.4 Web服务平台	239
11.2.1 SQL语言基本命令	189	12.5 其他网络用途	242

12.5.1 Web Forms	243	14.4 选择ClickOnce更新策略	263
12.5.2 Web Service	243	14.4.1 在应用程序启动后 检查更新	263
12.5.3 XML在VB.NET中的应用	243	14.4.2 在应用程序启动时 检查更新	264
第13章 程序调试和错误处理	245	14.4.3 强制执行更新	265
13.1 程序错误类型	245	14.4.4 指定更新时间间隔	265
13.1.1 编译错误	245	14.4.5 提供进行更新的用户界面	266
13.1.2 运行时错误	246	14.4.6 阻止更新检查	266
13.1.3 逻辑错误	247	14.5 手动部署ClickOnce应用程序	266
13.2 程序调试方法概述	248	14.5.1 先决条件	266
13.3 设置运行断点	248	14.5.2 使用Mage.exe命令行工具 部署应用程序	267
13.4 使用调试窗口	249	14.5.3 使用MageUI.exe图形工具 部署应用程序	268
13.4.1 立即窗口	249	14.6 ClickOnce部署清单	269
13.4.2 监视窗口	250	14.7 Windows Installer部署	271
13.4.3 本地窗口	250	14.7.1 部署基于Windows 的应用程序	271
13.5 单步调试和跳跃调试	250	14.7.2 可选的部署功能	272
13.6 异常和错误处理	251	14.8 Windows Installer使用示例	274
13.6.1 结构化异常处理	251	14.8.1 添加Samples文件夹	274
13.6.2 非结构化异常处理	252	14.8.2 为应用程序创建示例文件	274
13.6.3 何时使用结构化和 非结构化异常处理	254	14.8.3 将示例添加到安装程序中	274
13.7 处理异常的最佳做法	255	14.8.4 添加启动条件	274
13.8 如何避免错误	258	14.8.5 设置系统必备组件的属性	275
13.9 常见错误信息举例	258	14.8.6 设置部署项目的可选属性	275
第14章 部署Windows应用程序	260	14.8.7 在开发计算机上安装	275
14.1 选择部署策略	260	14.8.8 部署到其他计算机	275
14.2 ClickOnce部署	260	14.8.9 测试安装	276
14.2.1 ClickOnce部署的工作方式	261	参考文献	277
14.2.2 ClickOnce部署的功能	261		
14.3 选择ClickOnce部署策略	262		
14.3.1 从Web或网络共享安装	262		
14.3.2 从CD安装	262		
14.3.3 从Web或网络共享 启动应用程序	263		

第1章

Visual Basic 2010概述

Visual Basic 2010是从Visual Basic语言演变而来的，是一种为高效地生成类型安全和面向对象的应用程序而设计的语言。Visual Basic允许开发人员开发面向Windows、Web和移动设备的程序。与所有面向Microsoft .NET Framework的语言一样，使用Visual Basic编写的程序都具有安全性和语言互操作性方面的优点。

1.1 Visual Basic简介

Visual Basic简称VB，是Microsoft公司推出的一种Windows应用程序开发工具。是当今世界上使用最广泛的编程语言之一，它也被公认为是编程效率最高的一种编程方法。无论是开发功能强大、性能可靠的商务软件，还是编写能处理实际问题的实用小程序，VB都是最快速、最简便的方法。

何谓Visual Basic？“Visual”指的是采用可视化的开发图形用户界面（GUI）的方法，一般不需要编写大量代码去描述界面元素的外观和位置，而只要把需要的控件拖放到屏幕上的相应位置即可；“Basic”指的是BASIC语言，因为VB是在原有的BASIC语言的基础上发展起来的，至今包含了数百条语句、函数及关键词，其中很多和Windows GUI有直接关系。专业人员可以用Visual Basic实现其他任何Windows编程语言的功能，而初学者只要掌握几个关键词就可以建立实用的应用程序了。

VB提供了学习版、专业版和企业版，以满足不同的开发需要。学习版可使编程人员很容易地开发Windows和Windows NT的应用程序；专业版为专业编程人员提供了功能完备的开发工具；企业版允许专业人员以小组的形式来创建强健的分布式应用程序。

1.1.1 Visual Basic的诞生及发展

自Microsoft公司推出基于图形界面的Windows操作系统之后，越来越多的应用程序设计也趋向于图形化，但却苦于没有合适的开发工具。1991年，Microsoft公司展示了一款名为Thunder的新产品。该产品可以实现用鼠标“画”出所需的用户界面，再用简单的BASIC语言来编写业务逻辑，以此来生成一个完整的应用程序。该产品最终被定名为Visual Basic，它采用事件驱动、Quick Basic的语法和可视化的集成开发环境（Integrated Development Environment，简称IDE）。

虽然刚推出的版本缺陷很多，但经过Microsoft公司的不断努力，随后又相继在1992年、1993年、1995年和1997年推出了2.0～5.0的版本，1998年推出了6.0。整个过程中功能得到了不断完善和扩充，特别是Visual Basic 6.0较之以前的版本其功能和性能都大大增强了，尤其是数据库管理、网络应用方面更胜一筹。

推出Visual Basic 6.0后，Microsoft公司于2010年推出了完全支持面向对象的Visual Basic 2010，也为Visual Basic注入了新的活力。Visual Basic 2010的一个重要特点是公共语言运行时CLR，

CLR的目的是在同一个项目中支持不同的语言所开发的组件。所有CLR支持的代码都会被理解成为CLR可执行的机器代码然后再运行。2005年, Microsoft发布了Visual Basic 2005, 在该版本中基于的.NET框架为2.0。另外, 该版本的Visual Studio包含众多版本, 分别面向不同的开发角色, 同时还永久提供免费的Visual Studio Express版本。2007年Microsoft发布了Visual Studio 2008, 它提供了高级开发工具、调试功能、数据库功能和创新功能, 帮助开发人员在各种平台上快速创建当前最先进的应用程序。目前, Visual Basic的最新版本为Visual Basic 2010。它可以使开发人员在同一开发环境内创建面向多个.NET Framework版本的应用程序。

1.1.2 .NET Framework 4概述

.NET Framework是支持生成和运行下一代应用程序和XML Web Services的内部Windows组件。.NET Framework旨在实现下列目标。

提供一个一致的面向对象的编程环境, 而无论对象代码是在本地存储和执行, 还是在本地执行但在Internet上发布, 或者是在远程执行的。

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(包括由未知的或不完全受信任的第三方创建的代码)执行安全性的代码执行环境。

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。

使开发人员的经验在面对类型大不相同的应用程序(如基于Windows的应用程序和基于Web的应用程序)时保持一致。

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

.NET Framework具有两个主要组件: 公共语言运行库和.NET Framework类库。公共语言运行库是.NET Framework的基础。可以将运行库看做一个在执行时管理代码的代理, 它提供内存管理、线程管理和远程处理等核心服务, 并且还强制实施严格的类型安全, 以及可提高安全性和可靠性的其他形式的代码准确性。事实上, 代码管理的概念是运行库的基本原则。以运行库为目标的代码称为托管代码, 而不以运行库为目标的代码称为非托管代码。.NET Framework的另一个主要组件是类库, 它是一个综合性的面向对象的可用类型集合, 可以使用它开发多种应用程序, 这些应用程序包括传统的命令行或图形用户界面(GUI)应用程序, 也包括基于ASP.NET所提供的最新创新的应用程序(如Web窗体和XML Web Services)。

.NET Framework可由非托管组件承载, 这些组件将公共语言运行库加载到它们的进程中并启动托管代码的执行, 从而创建一个可以同时利用托管和非托管功能的软件环境。.NET Framework不但提供若干个运行库宿主, 而且还支持第三方运行库宿主的开发。

如图1.1所示为公共语言运行库和类库与应用程序之间, 以及与整个系统之间的关系, 并且还显示了托管代码如何在更大的结构内运行。

.NET Framework 4与使用.NET Framework早期版本生成的应用程序有很高的兼容性, 除了为了提高安全性、标准遵从性、正确性、可靠性和性能所做的一些更改之外。.NET Framework 4不能自动使用自己的公共语言运行时版本来运行由.NET Framework早期版本生成的应用程序。若要使用.NET Framework 4运行较早的应用程序, 则必须使用Visual Studio中项目的属性指定的目标.NET Framework版本编译应用程序, 或使用应用程序配置文件中的<supportedRuntime>元素指定所支持的运行时。

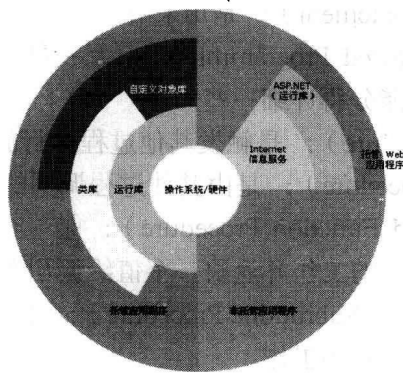


图1.1 .NET Framework 4.0环境

1.1.3 Visual Basic基本概念及常用术语

为了使读者更好地理解本书的内容，这里先介绍一下Visual Basic的一些常见基本概念及术语。

- 对象：面向对象程序设计的核心，是对现实世界中对象的模型化，是代码和数据的组合，具有自己的状态和行为。在进行具体的Visual Basic编程时，需以对象的眼光看待整个程序设计过程。Visual Basic中的对象主要是窗体和控件。
- 属性：对象中的数据，用于表示对象的状态，决定对象的外观。在Visual Basic中窗体和控件的大小、标题或颜色等都是其属性。
- 方法：对象中的代码决定对象可以进行的动作。方法具体指嵌入对象定义中的程序代码，用于定义对象怎样处理信息并响应某事件。在Visual Basic中控件的显示、隐藏动作等都是该控件的方法。
- 事件：由用户或操作系统引发的动作，如按键、单击鼠标、双击鼠标、一段时间的限制等。
- 类：用来创建对象的样板，包含所创建对象的状态描述和方法定义。

以上都是Visual Basic的基本概念，也是面向对象程序设计语言的基础概念。Visual Basic程序设计中较常用的几个术语如下。

- 项目 (Project)：是用户创建的文件集合，这个集合包括用户的Windows应用程序。
- 控件 (Control)：是Toolbox窗口中用户置于窗体上的工具，用于配合用户控制程序流程。
- 代码 (Code)：是所写的编程语句的另一个名字。
- 像素 (Pixel)：代表图形元素，表示监视器上最小的可寻址的图形点。
- 全局变量 (Global Variable)：就是在整个模块内或整个应用程序内均可使用的变量。
- 函数 (Function)：是一个例程，接受零个、一个或多个参数并根据这些参数返回一个结果。
- 死循环 (Infinite Loop)：是一个永不终止的循环。
- 语法错误 (Syntax Error)：是由于拼错一条命令或使用不正确的语法引起的一种错误。
- 消息框 (Message Box)：是为向用户提供信息而显示的对话框。
- 循环 (Loop)：就是一组重复执行的程序指令集。

- 赋值语句 (Assignment Statement)：是用来给控件、变量或其他对象赋值的程序语句。
- 结构化程序设计 (Structured Programming)：是一种程序设计方法，用它来把长程序分成几个小过程，尽可能分得详细一些。
- 调用过程 (Calling Procedure)：是触发其他过程执行的过程。
- 被调用过程 (Called Procedure)：是由其他过程调用的过程。
- 标准函数过程 (Standard Function Procedure)：是一个独立的非事件过程，当被其他过程调用时，它完成一定的工作并返回一个值给调用者。
- 标准子程序过程 (Standard Subroutine Procedure)：是一个独立的非事件过程，当被其他程序调用时，它完成一定的工作。
- 引用传递 (By Reference)：是一种传递值并允许被调用过程修改这些值的方法。它也叫做通过地址传递 (By Address)。
- 值传递 (By Value)：是一种传递值并保护调用过程的传递数据，因而被调用过程不能改变此数据的方法。
- 编辑掩码 (Edit Mask)：是一个格式字符串，如 “#，###，##”，它指定怎样显示数字和字符串数据。

1.2 Visual Basic 2010中的新增功能

1.2.1 自动实现的属性

“自动实现的属性”可以快速指定类的属性，而无须编写代码对该属性执行Get和Set操作。在为自动实现的属性编写代码时，除了创建关联的Get和Set过程之外，Visual Basic编译器还会自动创建一个私有字段来存储该属性变量。通过自动实现的属性，可在一行中声明一个包含默认值的属性。下面的示例演示了3个属性声明。

```
Public Property Name As String
Public Property Owner As String = "DefaultName"
Public Property Items As New List(Of String) From {"M", "T", "W"}
Public Property ID As New Guid()
```

自动实现的属性等同于将属性值存储在私有字段中的属性。下面的代码示例演示一个自动实现的属性。

```
Property Prop2 As String = "Empty"
```

下面的代码示例演示与上述自动实现的属性示例等效的代码。

```
Private _Prop2 As String = "Empty"
Property Prop2 As String
    Get
        Return _Prop2
    End Get
    Set(ByVal value As String)
        _Prop2 = value
    End Set
End Property
```

1.2.2 集合初始值设定项

“集合初始值设定项”提供了短语法，供创建集合并用一组初始值填充它。当从一组已知值创建集合时，集合初始值设定项十分有用。这组已知值可以是菜单选项、类别列表、一组初始数值、日期或月份名称之类字段串的静态列表或地理位置的静态列表（如用于验证目的的省/市/自治区列表）。

通过使用From关键字后跟大括号（{}），可以标识集合初始值设定项。这与数组（Visual Basic）中介绍的数组文本语法类似。下面的示例演示使用集合初始值设定项创建集合的各种方法。

```
' Create an array of type String().
Dim winterMonths = {"December", "January", "February"}
' Create an array of type Integer()
Dim numbers = {1, 2, 3, 4, 5}
' Create a list of menu options. (Requires an extension method
' named Add for List(Of MenuOption)
Dim menuOptions = New List(Of MenuOption) From {{1, "Home"},
                                                {2, "Products"}}
```

注意

集合初始值设定项由逗号分隔值列表组成，这些值用大括号（{}）括起，前面带有From关键字。

1.2.3 隐式行断续

Visual Basic是一种面向行的语言，它使用类似于英语的清晰语法来增强可读性。但这通常会导致代码遇到每行80个字符的限制，从而迫使开发人员要进行大量滚动。用户可以使用下画线字符来告知编译器应将下一行作为当前行继续处理（也就是说，将多个物理行视为单个逻辑行）。但不得不重复地输入下画线字符，一直很令人烦恼，而事实上多年以来排在首位的功能请求就是让编译器“解决这个问题”。

而在Visual Basic 2010中，编译器能够解决这个问题。编译器现在知道哪些标记（如逗号、圆括号和运算符）往往出现在行继续符前面，并且它会插入字符，因此开发人员不再需要插入字符。例如，用逗号作为Visual Basic语句的结尾肯定不合逻辑；编译器知道这一点，因此，当编译器看到诸如{comma, enter}这样的标记流时，它会推断出存在行继续符。表1.1是有关可以省略下画线字符的部分情况的列表。

表1.1 隐式行断续语法元素

语法元素	示例
在左括号（（）之后或 右括号（）之前	<pre>Dim username = GetUsername(Security.Principal.WindowsIdentity.GetCurrent().Name, CChar("\", 1)</pre>
在左大括号（{）之后或 右大括号（}）之前	<pre>Dim customer = New Customer With { .Name = "Terry Adams", .Company = "Adventure Works", .Email = "terry@www.adventure-works.com" }</pre>

语法元素	示例
在串联运算符 (&) 之后	<pre>cmd.CommandText = "SELECT * FROM Titles JOIN Publishers " & "ON Publishers.PubId = Titles.PubID " & "WHERE Publishers.State = 'CA'"</pre>
在赋值运算符 (=、&=、:=、+=、-=、*=、/=、\=、^=、<<=、>>=) 之后	<pre>Dim fileStream = My.Computer.FileSystem. OpenTextFileReader(filePath)</pre>

1.2.4 Lambda表达式

“Lambda表达式”是一种没有名称的函数或子例程，可在委托有效的任何地方使用。Lambda表达式可以是函数或子例程，且可为单行或多行。可以将值从当前范围传递到Lambda表达式。

使用Function或Sub关键字可以创建Lambda表达式，就像创建标准函数或子例程一样。但是，Lambda表达式包括在语句中。下面的示例是一个Lambda表达式，该表达式递增其参数并返回值。该示例同时显示了一个函数的单行和多行Lambda表达式语法。

```
Dim increment1 = Function(x) x + 1
Dim increment2 = Function(x)
    Return x + 2
End Function
' Write the value 2.
Console.WriteLine(increment1(1))
' Write the value 4.
Console.WriteLine(increment2(2))
```

Lambda表达式的语法类似于标准函数或子例程的语法，区别如下。

- Lambda表达式没有名称。
- Lambda表达式不能有修饰符，如Overloads或Overrides。
- 单行Lambda函数不使用As子句来指定返回类型。相反，类型是从Lambda表达式主体计算得出的值推断而来的。例如，如果Lambda表达式的主体为cust.City="London"，则其返回类型为Boolean。
- 在多行Lambda函数中，可以使用As子句指定返回类型，或者省略As子句以便推断返回类型。省略某多行Lambda函数的As子句时，返回类型被推断为该多行Lambda函数中所有Return语句的主导类型。“主导类型”是数组文本中所有其他类型都可以扩大到的唯一类型。如果无法确定此唯一类型，则主导类型是数组中所有其他类型都可以收缩到的唯一类型。如果这两种唯一类型都无法确定，则主导类型是Object。例如，如果为数组文本提供的值列表包含Integer、Long和Double类型的值，则得到的数组属于Double类型。Integer和Long都扩大到Double且仅扩大到Double。因此，Double是主导类型。
- 单行函数的主体必须是返回一个值的表达式，而不是语句。单行函数没有Return语句。单行函数返回的值是函数主体中的表达式的值。
- 单行子例程的主体必须是单行语句。

- 单行函数和子例程不包括End Function或End Sub语句。
- 可以使用As关键字指定Lambda表达式参数的数据类型，也可以推断参数的数据类型。要么所有参数都必须具有指定的数据类型，要么必须推断所有类型。
- 不允许使用Optional和Paramarray参数。
- 不允许使用泛型参数。

1.2.5 等价类型支持

现在可以部署已经嵌入类型信息（Type Information）的应用程序，而不必非从Primary Interop Assembly（PIA）导入类型信息。有了嵌入的类型信息，应用程序可以在运行时使用类型，而不需要引用运行时程序集（Runtime Assembly）。如果发布了不同版本的运行时程序集，包含嵌入类型信息的各版本程序都可以正常工作，而不需要重新编译。

1.2.6 共变和逆变

共变（Covariance）允许使用比泛型参数（Generic Parameter）更具派生的类型（More Derived Type），而逆变（Contravariance）允许使用更少派生的类型（Less Derived Type）。这实现了变体接口（Variant Interfaces）的类的隐式转换（Implicit Conversion），并且给具有变体委托类型的方法签名的匹配提供了更多的灵活性。用户可以用新关键字In和Out来创建变体接口和委托。 .NET 框架还给许多既有的泛型接口和委托引入变体支持，包括IEnumerable（Of T）接口和Func（Of TResult）、Action（Of T）委托。

1.2.7 定位到

通过使用“定位到”功能，可在解决方案中查找并定位到特定位置，也可以浏览解决方案中的元素。该功能可帮助从查询中选出一组适当的匹配结果，如图1.2所示。

可以使用“定位到”功能搜索符号或文件。首先在“编辑”菜单上单击“定位到”，在上部的文本框中，输入搜索字符串。如果搜索字符串包含大写字符，则搜索区分大小写；否则，搜索不区分大小写。



图1.2 “定位到”功能

注意

在输入时，结果将显示在下部的列表框中，随着输入更多的内容，列表框中的结果将相应变化。例如，如果输入Click，则会显示所有包含“Click”的符号名；如果再输入一个空格和字母E，则列表将进行筛选，只显示包含“Click”和“E”的符号名（空格视为and关系）。

搜索结果可能包含解决方案中的符号定义和文件名，但不包含命名空间或局部变量。双击某个结果可以在代码中查看其定义。通过使用“Camel大小写”格式和下画线字符，可以将符号分为多个关键字，这样可搜索包含在符号中的关键字。例如，要查“AddOrderHeader”符号，可以搜索“add”、“order”、“header”、“order add”、“AOH”及其他组合，如表1.2所示。