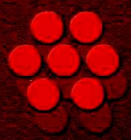
.NET Windows 应用开发教程



李勇平 编

兵器工业出版社

北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

.NET Windows 应用开发教程

李勇平 编

兵器工业出版社
北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

内 容 简 介

本书共分成 4 个部分，全面介绍了.NET Windows 应用开发。其中，第一部分：.NET 基础与 C＃编程技术，主要介绍了.NET 的概念、.NET 构成、C#编程基础以及面向对象的编程技术。该部分包含本书的第 1 章到第 5 章。第二部分：.NET Windows 应用开发技术，主要介绍 Windows 窗体、控件、事件模型、GDI+以及应用程序的调试、测试和部署。该部分包含本书的第 6 章到第 10 章。第三部分：.NET 数据访问技术，主要包括 SQL Server 数据库基础、ADO.NET、Windows 窗体数据绑定技术以及.NET I/O 技术。该部分包含本书的第 11 章到第 14 章。第四部分：.NET 组件技术，主要包括.NET 组件、自定义控件等相关技术。该部分主要包括第 15 章、第 16 章。

本书内容实用、循序渐进，针对每一个知识点都按照为什么学、这个知识点的含义、如何应用这个知识点的思路进行讲解，让读者知其然并知其所以然。

本书适合初中级 Windows 应用开发人员、大中专院校的学生。另外，还可以作为微软 MCAD 考试的辅导教材。

书中部分实例请从 http://www.b-xr.com 下载。

图书在版编目（CIP）数据

.NET Windows 应用开发教程/李勇平编. —北京：兵器工业出版社；北京希望电子出版社，2004.3
ISBN 7-80172-164-0

Ⅰ.N… Ⅱ.李… Ⅲ.计算机网络—程序设计—教材 Ⅳ.TP393

中国版本图书馆 CIP 数据核字（2003）第 113012 号

出　　版：兵器工业出版社 北京希望电子出版社
邮编社址：100089 北京市海淀区车道沟 10 号
　　100080 北京市海淀区知春路甲 63 号卫星大厦 3 层
发　　行：北京希望电子出版社
电　　话：（010）62520290（发行）（010）62532258（门市）
经　　销：各地新华书店 软件连锁店
印　　刷：北京双青印刷厂
版　　次：2004 年 3 月第 1 版第 1 次印刷

封面设计：梁运丽
责任编辑：李亚明 宋丽华 周 艳
责任校对：但明天
开　　本：787×1092 1/16
印　　张：38.5
印　　数：1-4000
字　　数：893.6 千字
定　　价：52.00 元

前　言

自 2000 年 6 月微软宣布自己的.NET 战略以来，.NET 技术一直受到业界的广泛关注。2002 年 3 月 Visual Studio.NET 在中国大陆发布，2003 年 5 月 Visual Studio.NET 2003 在中国大陆发布，这一系列的事实都表明.NET 已经由战略变为现实了。

Visual Studio.NET 是一个功能强大、高效并且可以扩展的编程环境，是一个构建企业分布式应用的开发平台，基于 XML Web 服务的技术得到了 Visual Studio.NET 的大力支持。使用 Visual Studio.NET 开发平台可以构建 Windows 应用程序、Web 应用程序以及 Web 服务。使用 Visual Studio.NET 构建分布式应用程序将更加简单、高效而且稳定。

在.NET 技术发展之际，希望电子出版社组织编写了本书，介绍.NET Windows 的应用开发。

本书有以下特点：

- 实用。本书的作者有丰富的开发经验，都是长期从事.NET 技术研究的专家、学者。本书的指导原则就是实用性。每一个知识点都是大量工程实践中总结出来的宝贵经验。无论你是.NET 的初学者还是有一定基础的开发者，你都会从本书中获得宝贵的经验。
- 循序渐进。本书设计的第 2 个指导原则就是循序渐进，由易到难。
- 遵循人的认知规律。我们在组织每一个知识点的时候都按照为什么学这个知识点，这个知识点的具体含义是什么，如何应用这个知识点的思路来进行的。我们的目的是让读者知其然并知其所以然。

本书共分成 4 个部分。这 4 个部分分别如下：

第一部分：.NET 基础与 C＃编程技术，该部分主要介绍了.NET 的概念、.NET 构成、C#编程基础以及面向对象的编程技术。该部分包含本书的第 1 章到第 5 章。

第二部分：.NET Windows 应用开发技术，该部分主要介绍 Windows 窗体、控件、事件模型、GDI+以及应用程序的调试、测试和部署。该部分包含本书的第 6 章到第 10 章。

第三部分：.NET 数据访问技术，该部分主要包括 SQL Server 数据库基础、ADO.NET、Windows 窗体数据绑定技术以及.NET I/O 技术。该部分包含本书的第 11 章到第 14 章。

第四部分：.NET 组件技术，该部分主要包括.NET 组件、自定义控件等相关技术。该部分主要包括第 15 章、第 16 章。

本书主要由李勇平编著，另外参与本书编著的人员还有万丽、宋乃辉、靖琦忠、曹东波、李海红、宴菊秀、蒲小波、李艳红、康宏琳等。

本书出版过程中，得到兵器工业出版社、北京希望电子出版社各位领导和工作人员的支持和帮助，在此表示衷心的感谢。另外，还要特别感谢我的父母和曹东波先生，是你们的鼓励让我能够集中精力完成本书的编著。

需要本书或技术支持的读者，请与北京中关村 083 信箱（邮编 100080）发行部联系，电话：010-62528991，62524940，62521921，62521724，82610344，82675588（总机）传真：010-62520573，E-mail：yanmc@bhp.com.cn

作者

2003 年夏

目　录

第一部分　.NET 基础与 C# 编程技术

第 1 章　.NET 平台与 C#概述 1
1.1　.NET 概述 1
1.1.1　.NET 简介 1
1.1.2　.NET 框架（.NET Framework）简介 5
1.1.3　公共语言运行库 6
1.2　C#简介 8
1.2.1　C#编程环境 8
1.2.2　第 1 个 C#程序 10
1.2.3　C#程序结构 12
1.3　C# 语言基础 15
1.3.1　变量和数据类型 15
1.3.2　运算符和表达式 17
1.3.3　类型转换 18
1.3.4　枚举类型 20
1.3.5　结构类型 21
1.4　各种语句结构 23
选择语句 23
1.5　实例分析 31
1.5.1　程序分析说明 31
1.5.2　代码编写调试 31
1.5.3　程序测试总结 34
1.6　小结 34
第 2 章　面向对象编程基础 35
2.1　面向对象基本概念 35
2.1.1　对象的概念 35
2.1.2　类的概念 36
2.1.3　类的基本要素 37
2.1.4　类的基本特征 38
2.1.5　面向对象编程的特点和优点 40
2.2　创建类 41
2.2.1　字段 41
2.2.2　方法 43
2.2.3　创建和使用对象 46
2.2.4　属性 48
2.3　对象的构造和析构 53
2.3.1　对象的构造和析构 53
2.3.2　static 关键字 56
2.3.3　this 关键字 57
2.4　通用类型系统 58
2.4.1　简介 58
2.4.2　装箱操作 59
2.4.3　取消装箱操作 60
2.5　实例研究 61
2.5.1　程序分析说明 61
2.5.2　代码编写和调试 62
2.5.3　程序测试和总结 66
2.6　小结 67
第 3 章　数组、日期和字符串 68
3.1　数组 68
3.1.1　一维数组 68
3.1.2　多维数组 70
3.1.3　交错数组 73
3.1.4　Array 类简介 75
3.1.5　数组作为方法的参数 77
3.2　日期和时间数据 78
3.2.1　DateTime 结构 78
3.2.2　TimeSpan 类 80
3.3　字符串类 83
3.3.1　String 类 83
3.3.2　StringBuilder 类 88
3.3.3　格式化数据 93
3.3.4　字符串类型转换成其他数据类型 97
3.4　实例分析 98
3.4.1　程序分析说明 98
3.4.2　代码编制 98
3.4.3　实例测试总结 101
3.5　小结 102
第 4 章　面向对象编程进阶 103

4.1 重载.....103
4.1.1 方法重载.....103
4.1.2 操作符重载.....107
4.2 类的继承性.....113
4.2.1 继承的含义.....114
4.2.2 方法重写.....123
4.2.3 base 关键字.....123
4.2.4 protected 关键字.....124
4.2.5 密封类以及密封方法.....129
4.3 类的多态性.....130
4.3.1 虚方法.....130
4.3.2 抽象类和抽象方法.....138
4.4 实例研究.....142
4.4.1 程序分析.....143
4.4.2 代码编写和调试.....143
4.4.3 程序测试和总结.....146
4.5 小结.....147
第 5 章 命名空间与异常处理.....148
5.1 命名空间.....148
5.1.1 命名空间的概念.....148
5.1.2 命名空间的定义和使用.....149
5.1.3 嵌套命名空间.....152
5.2 System 命名空间.....153
5.2.1 Math 类.....154
5.2.2 Random 类.....156
5.3 System.Collections 命名空间.....158
5.3.1 ArrayList 类.....159
5.3.2 IEnumerator 接口.....160
5.3.3 Hashtable 类.....163
5.4 异常处理.....168
5.4.1 异常处理结构.....168
5.4.2 finally 关键字.....173
5.4.3 System.Exception 类.....174
5.5 实例分析.....177
5.5.1 程序分析说明.....177
5.5.2 代码编写与调试.....177
5.5.3 测试与总结.....183
5.6 小结.....184

第二部分 .NET Windows 应用开发技术

第 6 章 Windows 窗体设计.....185
6.1 Visual Studio.NET 开发环境介绍.....185
6.1.1 起始页.....185
6.1.2 使用开发环境开发应用程序实例 1.....186
6.1.3 使用开发环境开发应用程序实例 2.....190
6.1.4 自定义开发环境.....193
6.2 Windows 程序设计.....194
6.2.1 启动界面制作.....194
6.2.2 简单计算程序设计.....201
6.3 窗体对象详解.....204
6.3.1 常见属性.....205
6.3.2 常见方法.....210
6.3.3 事件介绍.....211
6.3.4 System.Windows.Forms 命名空间简介.....215
6.4 实例分析.....218
6.4.1 程序分析说明.....218
6.4.2 程序编写.....219
6.4.3 程序测试和总结.....222
6.5 小结.....223
第 7 章 Windows 常用控件.....224
7.1 控件对象介绍.....224
7.1.1 控件简介.....224
7.1.2 控件基本的属性设置.....226
7.1.3 控件的基本方法和事件简介.....231
7.2 各类控件的使用.....233
7.2.1 值设置控件.....233
7.2.2 从列表中选择控件.....239
7.2.3 分页控件.....244
7.2.4 Timer 控件.....247
7.2.5 菜单、工具栏、状态栏.....249
7.2.6 Windows 内置对话框.....254
7.3 计算器程序的编写.....259
7.3.1 程序分析说明.....259
7.3.2 界面设计和代码编写.....260

7.3.3 程序测试和总结.....265
7.4 小结.....265
第 8 章 基本界面布局和设计.....266
8.1 Windows 窗体布局概述.....266
8.1.1 基本窗体布局.....266
8.1.2 程序设计实例——多窗口启动实例.....268
8.2 对话框的使用.....272
8.2.1 模式对话框.....272
8.2.2 无模式对话框.....276
8.3 多文档界面设计.....279
8.3.1 创建多文档程序.....279
8.3.2 创建多文档程序实例-文本编辑程序编辑.....285
8.4 资源管理器样式界面设计.....293
8.4.1 树状控件.....293
8.4.2 列表控件.....295
8.4.3 水平调整控件之间的尺寸.....297
8.4.4 资源管理器样式程序设计实例.....300
8.5 小结.....302
第 9 章 GDI+编程.....303
9.1 GDI+绘图的基本概念.....303
9.1.1 GDI+概述.....303
9.1.2 基本绘图表面概述.....304
9.1.3 Graphics 类和 Graphics 对象的创建.....306
9.1.4 坐标以及坐标变换.....309
9.2 基本的 GDI＋对象.....317
9.2.1 画笔.....317
9.2.2 笔刷.....318
9.2.3 颜色.....321
9.2.4 字体.....322
9.3 基本图形的绘制.....324
9.3.1 线条和形状.....324
9.3.2 绘制文本.....328
9.3.3 绘制图像.....328
9.4 Windows 打印功能的实现.....330
9.4.1 创建打印作业实现打印工作.....331
9.4.2 页面设置.....332
9.4.3 打印机设置.....333
9.4.4 打印预览.....334
9.5 实例分析.....338
9.5.1 程序分析说明.....338
9.5.2 程序编制和调试.....338
9.6 小结.....342
第 10 章 应用程序调试、测试和部署.....343
10.1 应用程序调试.....343
10.1.1 错误类型.....343
10.1.2 调试版和发布版.....344
10.1.3 应用程序执行的方式.....345
10.1.4 调试窗口.....351
10.1.5 .NET Diagnostics 技术.....353
10.2 应用程序测试.....357
10.2.1 测试概述.....357
10.2.2 测试方法.....359
10.2.3 测试案例.....361
10.3 应用程序部署.....362
10.3.1 部署基本概念.....362
10.3.2 创建部署项目.....363
10.4 小结.....371

第三部分 .NET 数据访问技术

第 11 章 SQL Server 数据库简介.....372
11.1 数据库基本概念.....372
11.1.1 数据库概念.....372
11.1.2 关系数据库概念.....375
11.1.3 样本数据库——Northwind 数据库简介.....377
11.2 设计一个数据库.....378
11.2.1 设计步骤.....378
11.2.2 用 SQL Server 企业管理器创建数据库.....379
11.2.3 使用 Access 设计数据库.....383
11.3 数据库查询语言.....384
11.3.1 SQL 简介.....384
11.3.2 查询语言综述.....384
11.3.3 多表查询.....388

11.3.4 数据操作语言.....389
11.4 存储过程和视图.....390
11.4.1 存储过程.....390
11.4.2 视图.....393
11.5 小结.....396
第 12 章 ADO.NET 数据访问技术.....397
12.1 .NET 数据访问概述.....397
12.1.1 System.Data 命名空间.....398
12.1.2 ADO.NET 体系结构.....398
12.1.3 ADO.NET 与 ADO.....402
12.2 使用.NET 数据提供程序访问数据...404
12.2.1 连接对象.....404
12.2.2 命令对象.....405
12.2.3 DataReader 类.....411
12.2.4 DataAdapter 类（数据适配器）.....414
12.2.5 实例研究.....415
12.3 创建和使用 DataSet.....417
12.3.1 数据表 DataTable.....418
12.3.2 表间关系 DataRelation.....426
12.3.3 使用数据适配器填充 DataSet.....430
12.3.4 使用数据集 DataSet 操纵数据库.....431
12.4 数据视图.....440
12.5 编写数据提供程序.....442
12.6 小结.....443
第 13 章 Windows 数据访问.....444
13.1 数据绑定概述.....444
13.1.1 数据绑定的必要性.....444
13.1.2 数据绑定实例.....453
13.1.3 数据绑定应用范围.....458
13.2 数据绑定原理.....459
13.2.1 Windows 窗体数据提供者.....459
13.2.2 数据使用者.....463
13.3 数据集内筛选和排序.....465
13.3.1 在数据表中直接筛选和排序.....466
13.3.2 使用数据视图.....467
13.3.3 程序实例.....473
13.4 在窗体上操作数据.....475
13.5 数据绑定应用综述.....481
13.5.1 查找表实例.....481
13.5.2 主控/详细关系实例.....485
13.6 小结.....488
第 14 章 .NET I/O 操作与文件管理.....489
14.1 流的概念.....489
14.1.1 .NET 流的体系结构.....489
14.1.2 Stream 类.....491
14.2 字节流.....492
14.2.1 FileStream 类.....493
14.2.2 BufferedStream 类.....501
14.2.3 MemoryStream 类.....502
14.3 字符数据输入和输出.....502
14.3.1 使用字节流处理 Unicode 字符数据的输入和输出.....503
14.3.2 TextReader 类和 TextWriter 类.....506
14.3.3 StreamReader 类和 StreamWriter 类.....507
14.3.4 StringReader 类和 StringWriter 类.....512
14.4 二进制数据的输入和输出.....513
14.4.1 使用字节数据流处理二进制数据的输入和输出.....513
14.4.2 BinaryReader 类和 BinaryWriter 类.....516
14.5 文件的随机访问技术.....520
14.6 文件系统.....522
14.6.1 文件管理.....522
14.6.2 目录管理.....526
14.6.3 路径.....527
14.6.4 程序实例.....528
14.7 小结.....533

第四部分 .NET 组件技术

第 15 章 .NET 组件技术.....534
15.1 再谈属性.....534
15.1.1 索引器.....534
15.1.2 索引属性（访问对象的属性）..540

15.2 委托和事件.....543
15.2.1 委托.....543
15.2.2 事件.....545
15.2.3 .NET 事件模型.....549
15.3 接口与组件.....556
15.3.1 组件.....556
15.3.2 接口.....557
15.4 创建.NET 组件.....562
15.4.1 用于设计组件的接口和类.....562
15.4.2 创建和使用组件.....564
15.5 使用组件技术实现 windows 窗体继承.....569
15.5.1 窗体继承的概述.....569
15.5.2 创建和使用继承窗体实例.....571
15.6 小结.....576
第 16 章 自定义控件.....577
16.1 创建自定义控件.....577
16.1.1 创建自定义控件的方法.....577
16.1.2 创建自定义控件.....578
16.1.3 从 System.Windows.Forma.Control 派生自定义控件.....579
16.1.4 控件的绘制.....583
16.2 从现有控件派生自定义控件.....585
16.2.1 创建和使用派生自定义控件实例.....
16.2.2 自定义控件的属性、方法和事件.....591
16.3 复合自定义控件（用户控件）.....592
16.3.1 创建和使用复合自定义控件（用户控件）实例.....592
16.3.2 实现设计时特性.....600
16.5 小结.....602

第一部分　.NET 基础与 C# 编程技术

第 1 章　.NET 平台与 C#概述

微软公司 2002 年推出了.NET，那么到底.NET 是什么呢？本章将对.NET 平台做简单的介绍，并且将讲解.NET Framework 的体系结构。C#是为.NET 平台度身定做的新的编程语言，本章将介绍 C# 的编程环境，C# 的程序结构以及 C# 语言的基本编程概念，包括变量、数据类型、表达式以及各种语句结构。

1.1　.NET 概述

本节将告诉你到底什么是.NET。本节首先对.NET 做了一个简单的介绍，然后介绍了.NET 开发平台，在介绍.NET 开发平台时，将重点介绍.NET Framework 的相关知识。

1.1.1　.NET 简介

随着网络经济的到来，微软公司希望帮助用户能够在任何时候、任何地方、利用任何工具都可以获得网络上的信息，并享受网络通信所带来的快乐。.NET 战略就是为着实现这样的目标而设立的。

网络正以不可想象的速度发展着。最初的网络服务商向客户提供信息，各个网络服务商之间缺乏紧密的联系，它们就像一个个孤岛。图 1-1 说明了这种网络。

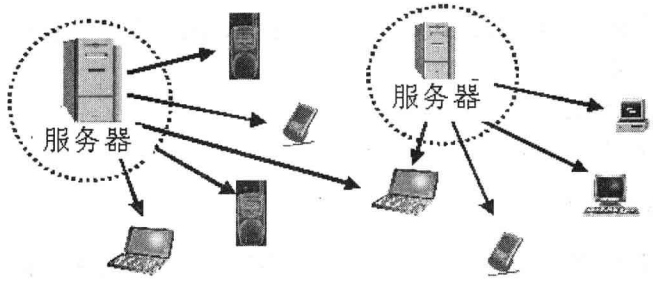


图 1-1　目前的网络结构

网络服务商不仅向客户提供数据，而且还向客户提供图片。当客户向 Web 服务器请求数据时，Web 服务器会从数据库中查找数据，然后把返回的信息以 HTML 的形式返回客户浏览器。图 1-2 说明了这个过程。在这种网络模式中，当用户需要对检索的数据重新过滤或者排序时，客户必须重新向 Web 服务器发送请求。

电子技术不断发展，用户使用的设备有掌上电脑、智能电话、WebTV、书写板电脑等。用户有了很多电子设备，分布在不同的位置：电脑可能在家里，掌上电脑、手机可能随身携带。用户希望不管在哪里，不管使用什么设备都可以访问同样的信息。在家的时候想用电脑听 MP3，在公司的时候想用手机看看家里电脑上的通讯录的信息。也就是用户有在任

何地点通过任何设备访问信息的需要，用户希望拥有自己的个人信息空间，有“所有信息都归我所有的感觉”。

对于网络服务商来说，要满足用户的需求，各个网络服务商不应该再是个孤岛了，所有的网络服务商形成一个整体，共同为用户提供信息，如图 1-2 所示。由于整个网络是个整体，用户在存取网络提供的服务的时候就感觉不到网络的存在了。

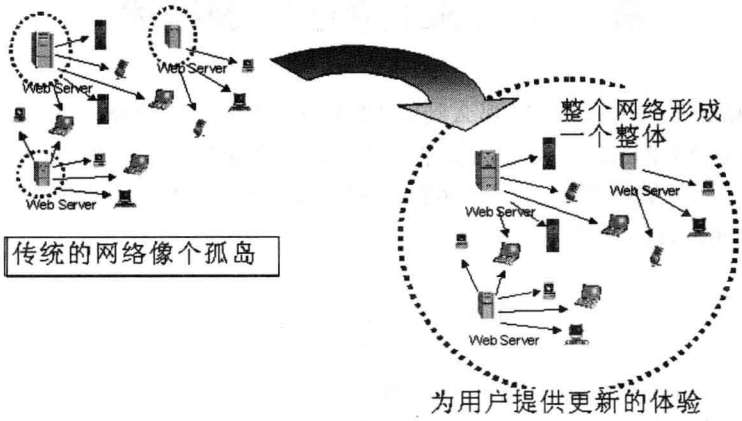


图 1-2 .NET 带来的变化

微软为了满足互联网的发展，率先推出了新一代互联网开发工具.NET，以进一步巩固其在世界软件业的地位。

.NET 的核心组件如下：

- .NET Framework（.NET 框架）。
- .NET 企业服务器。
- .NET Web 服务，一组用于创建互联网操作系统的构建块，其中包括 Passport.NET（用于用户认证）以及用于文件存储的服务、用户首选项管理、日历管理以及众多的其他任务。
- Visual Studio.NET。

下面来看一下.NET 的推出的巨大意义。

1. 提高开发者的效率

Microsoft .NET 的策略是将互联网本身作为构建新一代操作系统的基础，对互联网和操作系统的设计思想进行合理延伸。这样，开发人员必将创建出独立于设备和平台的应用程序，以便轻松实现互联网连接。.NET 无疑是通向网络时代的一个非常重要的里程碑。

.NET 对开发人员来说也十分重要，因为它不但会改变开发人员的开发应用程序的方式，而且使得开发人员能创建出全新的各种应用程序。开发者能够使用.NET 快速地创建独立于设备，独立于平台的应用程序，包括 Windows 应用程序、Web 应用程序以及 Web 服务。

（1）统一的编程模式

让我们来看看 Windows 应用是如何编写的。回溯到 10 年甚至 15 年，写一个 Windows 应用，实际上只有一种方式来做。首先，你会启动 C 编译器，并且在程序中包含“Windows.h”

文件，然后你会写一些消息处理函数，来完成一个 Windows 程序的设计。这些工作都是非常复杂的，而且工作效率低下。

随着时间的推移，有了一些不同的编程模型。例如，Visual Basic 开创了一种快速开发方式，利用窗体来设计程序。用这种方式，可以从一个工具箱上拖放“控件”放到一个窗体中，然后，你可以写一些业务处理过程，代码都放在后台。但 C++走了另外一条道路，利用 MFC 与 ATL，使你编程更加灵活，功能更加强大，但它没有 Visual Basic 使用起来容易。

随着互联网的发展，不断有新的编程模型出现，如服务器端的 ASP 编程模型。

但是我们还在发展，常常因为你不经意间选择的编程模型决定你的编程语言，这一切都在束缚着你。

另外一个问题是，每种编程模型都对通用的问题有着不一样的解决方案。例如，在不同的编程模型中，对文件的输入输出，就有不同的处理方式；对内存的管理，也有不同的方式。

现在，.NET 框架所作的事情，就是统一这些编程模型，提供一种一致的 API。这大大简化了编程。

首先，.NET 框架提高了抽象层。我们来看看 COM，这是一个非常低级的二进制标准。如果使用 C++来进行 COM 编程，它是可怕的。你不得不处理 HRESULT 与 GUIDS，以及增加 refs 与 release，直到到处都有处理 bugs 的代码。在今天来看，COM 并不足够丰富，而且 COM 不支持核心的面向对象的概念。

.NET 框架使你远离这些麻烦，.NET 完全自动地去处理这些事情，让你可以从这些复杂的工作中解脱出来，并集中思想到运算规则以及业务逻辑上。

那么，.NET 框架是如何统一编程系统的？

在.NET 框架中，每件事都可以看作是一个对象，甚至一个 int 也可以看作是一个对象。没有变量，所有都是对象，所以就没有在 Visual Basic 中变量与对象的不同的说法。.NET 提供了面向对象的编程或软件组件的编程方式。例如，属性、方法、事件都是.NET 框架中的第一级的结构。如果你用 C++来写一个 ActiveX 控件，不得不用一些宏来定义属性以及事件。但在.NET 框架中，.NET 框架的开发语言不需要使用宏就可以直接定义属性和事件了。

另外，再来看看.NET 组织这些 API 的方式。过去人们用 Win32 API，实际上是一个巨大的扁平的 API，那里根本没有任何组织。我们对这 30 000 个 API 入口最好的办法就是按照字母排序。在.NET 框架中，所有一切都被组织在命名空间中。所以，如果看到一些关于输入输出的东西，可以去查看 System.IO 命名空间，然后会找到叫“文件”的类库。使用命名空间可以很容易地找到所需要的类库。

.NET 通过把 API 封装在命名空间中，形成框架类库，使得抽象层向上移，并且新的 API 更加容易使用。这样，程序员只需要关注所要做的事情，也就是说不需要完全懂得所有的 API 也能够写最简单的应用程序。

（2）多语言支持

.NET 框架是一个多语言的平台。通过这个平台，可以利用我们熟悉的编程语言、已有编程经验和已编写的代码来编写程序。另外一个关键是，.NET 框架中所支持的语言都是一流的。任何 API 发布到.NET 框架中，都可以立即被 C# 或任何在这一平台上实现的编程语言来使用。在.NET 中，我们可以用一种语言编写一个类，然后在另外一个语言中继承，还

可以在第 3 种语言中实例化。这种集成能力可以如此之深。当然，这也意味着你可以高度整合这些开发工具。这实际上只需要一个集成的开发环境，一个 Debugger 来使用这些语言。

现在，.NET 提供了 5 种语言：VB、C++、C#、J# 与 JavaScript 或者 JScript。但是在业界和学术界，有很多其他语言已被开发，或者正在被开发，包括 PASCAL、Fortran 等。

（3）简化应用程序的安装和部署

安装应用程序时经常碰到版本冲突的问题，这种问题称为“DLL 地狱”，严重时会造成注册表瘫痪和不同版本类库的错位。由于.NET 的元数据使程序集具有自我描述性，从而很好地克服了应用程序安装时所遇到的版本冲突问题。在.NET 框架中，绝对不需要为了运行而注册。我们可以生成一个子目录，把应用拖到目录里，然后运行它。当运行结束，可以删除目录，不管在什么机器上运行，都不会留下痕迹。

在.NET 中，可以在相同应用里，在机器上运行与安装不同的版本代码。我们可以有应用版本 1.0 和应用版本 2.0，并且可以并行运行它们。我们甚至可以在相同的进程中运行从这些应用中分离出来的对象。这也就意味着，我们不再有那些 DLL hell 问题。在.NET 框架中，应用程序将会根据版本而运行。所以，正像前面所说的，我们可以有一些数据库支持的库版本为 1.0、1.1 与 2.0，各种不同的应用可以继续使用不同的版本，因此，在默认情况下，当在机器中安装了新的应用时，原有应用也不会中断。

（4）Web Service 思想

Web Service 充分利用了 Web。Web Service 有着与创建 HTML 应用相同的架构。所有在网络上传递的一切都是通过 HTTP 协议。惟一改变的就是 Web Service 可以在 HTTP 基础上传递 XML。这样，我们就可以很好地利用以前创建的 Web 应用，就可以使原来的 Web 应用变成 Web Service。

XML Web Service 可以在异构系统集成。我们不需要应用的两端都运行在相同的 OS 上，甚至不需要应用的两端都用相同的中间件。有了 Web Service，应用的一端可能是在.NET 框架上，应用的另一端可能是运行在 WebLogic Server 上，运行着 JAVA。

Web Service 允许创建真正的可扩展的应用。传统的分布式应用技术都用了所谓远程调用的技术，如：DCOM、CORBA 等。这些技术使得这个世界看起来像是面向对象的，有一些对象运行在这台机器上，有一些对象运行在另外一台机器上。但是，这个架构只有在内部处理通讯，在一台机器上，或在一个本地的内部网络中，它可以执行得很好。但当开始扩展应用时，比如客户端在上海，而服务器在北京，就会出现问题。因为这种旧的技术基本上是不能够足够分布的。如果对象运行在一台机器上，应用程序会运行得很好。但是，当要把对象分布到无限的网络上，要与东海岸对话，需要了解对方对象的状况时，这就会出现问题。因为当使用这些技术提出一个调用时，我们需要等很久才能得到响应。这样，我们不得不持续保持呼叫状态，但这样会影响到系统可靠性。

XML Web Service 关键的就是使用了简单的开放的标准，这些标准有广泛的业界支持，而不是微软制定的标准。这些标准是微软与业界合作伙伴以及与标准组织如 W3C 合作创建的。

2. .NET 的用户体验

.NET 对最终用户来说非常重要，因为计算机的功能将会得到大幅度提升，同时计算机操作也会变得非常简单。特别地，用户将完全摆脱人为的硬件束缚：用户可以自由冲浪于

互联网的多维时空，而不是束缚在便携式电脑的方寸空间——可通过任何桌面系统、任何便携式电脑、任何移动电话或 PDA 进行访问，并可对其进行跨应用程序的集成。

.NET 可使用户轻松进行互联网连接，并轻松完成那些在当今看来十分费时而且费力的事务，它们往往要求用户进行数据重输入并需运行几个小时才能完成。通过将多项安全数据流合并到单一的用户界面（或者甚至是可编程决策引擎），.NET 架构将用户从充斥于当今 Web 的数据竖井的束缚中解脱出来。用户可以自由访问、自由查看、自由使用他们的数据。

3. .NET 对 IT 企业的重要意义

.NET 平台将从根本上改善计算机和用户之间进行交互的方式，最大限度地发挥电子商务中计算技术的重要作用。首先，让我们来分析一下当前商务计算世界的现状。

假定网站 A、B、C、D 都从事网上书店业务，作为消费者在买书的时候总是希望货比三家。那么怎么比较这些价格信息呢？通常的做法是进入 A、B、C、D 4 个网站，然后浏览相关的 HTML 信息，从而进行比较。现在有一家网站 F 想从事这样的服务，F 网站帮助消费者比较各个网上书店的信息，这样消费者只要浏览 F 网站就可以。网站 F 要达到这个目的必须从网站 A、B、C、D 获得相关的数据，如果网站 A、B、C、D 以 HTML 的形式向网站 F 提供数据，那么网站 F 就很难获得数据的真正含义，这样的比较就会很困难。网站 F 的开发也将是非常困难的。.NET 提供的 Web Service 将使这种模式的电子商务的开发、部署和实施都变得相当简单。这样，更多的类似于 F 这样的网站将能很快的构建，这对 IT 企业将是一个非常好的契机。

.NET 把雇员、客户和商务应用程序整合成一个协调的、能进行智能交互的整体，而各公司无疑将是这场效率和生产力革命的最大受益者。

简言之，.NET 将为人类创造一个崭新的电子商务世界。

1.1.2　.NET 框架（.NET Framework）简介

图 1-3 为.NET 框架图。从图中可以看出，.NET 框架由以下几部分组成。

- 公共语言运行库：在操作系统之上有个“公共语言运行库”（Common Language Runtime）。这是一个丰富的运行环境。这个运行环境提供了所有核心的服务，比如内存管理、无用单元回收、安全性等等。它是围绕着通用类型系统而建立的，并且为所有语言定义了通用类型系统。
- 基类库：在公共语言运行库上面是一组基类库，它提供一组最通用的功能，比如文件输入输出、串处理、访问 TCP/IP 等等。这些类可以为.NET 其他类继承和扩充。
- 可扩充类：在基类库之上，由数据访问类 ADO.NET 和 XML 相关的类，这是下一代的支持 XML Web 服务的数据访问技术。在类库上面还有 2 个框架：服务器端的框架和客户端的框架。在服务器端，有 ASP.NET 即 Web 应用服务器。Web Forms 是一个在 ASP.NET 里编写 Web 窗体的编程模型。Web Forms 支持 Web 服务与移动互联网工具集，并且允许为移动设备创建 UI。在客户端，有个 Windows Forms，可以把它当作 Visual Basic 窗体与 MFC 放在一起的一种革新。在 Windows 编程世界里，Windows Form 将最优化的两种特征兼顾。

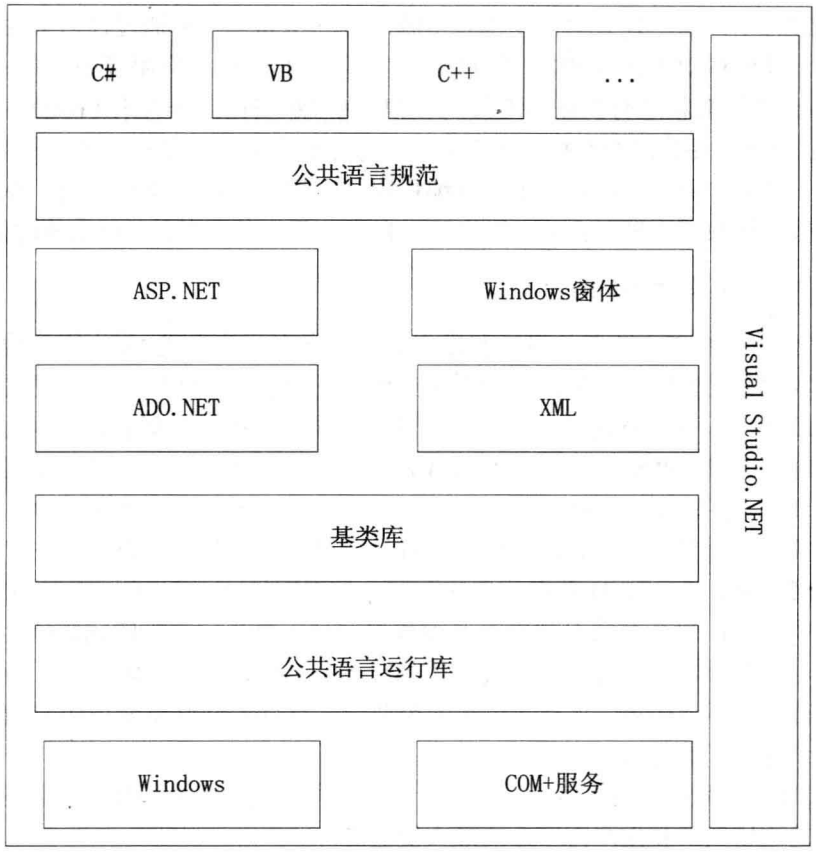


图 1-3 .NET Framework 体系结构

- 公共语言规范：它是一个标准，是所有的语言必须支持的一个最小的功能集合。公共语言规范通过说明一组与.NET 相兼容的语言必须遵守的规则，来定义.NET 所需要的规则，以便于所有语言可以运行在.NET 公共语言运行库之上。这些规则中有一条就是语言必须遵守通用类型系统。
- 多编程语言：.NET 提供了支持公共语言规范的 4 种语言：Visual Basic、C++、C# 与 JavaScript。但 IT 的合作伙伴正在创建支持.NET 公共语言运行库的其他语言，到目前，已经有超过 20 种的语言支持.NET 公共语言运行库。当然，支持所有功能特性的是 Visual Studio.NET。

1.1.3 公共语言运行库

本节将介绍公共语言运行库。

1. 公共语言运行库的特点

- 多语言支持

CLR 支持 C#，C++，VB.NET，J#，Perl，Cobol，Python，Fortran 等多种语言进行编程。多种语言的支持将使几乎所有的程序员都可以使用.NET 平台开发自己需要的程序。

● 公共类型系统

.NET 公共类型系统 (CTS) 是一套新型的公共数据类型，该公共数据类型在 CLR 中已被定义。CTS 定义了所有的标准类型，如 int、float、double 等。有了公共类型系统，从一种语言调用另外一种语言再也不需要特殊类型转换或调用规范。

2. 公共语言运行库运行过程

公共语言运行库的运行过程为：各种语言源程序经过一次编译为微软中间语言 MSIL，MSIL 经过 2 次编译为可执行代码，如图 1-4 所示。从中间语言到可执行代码的过程是由公共语言运行库来完成的。

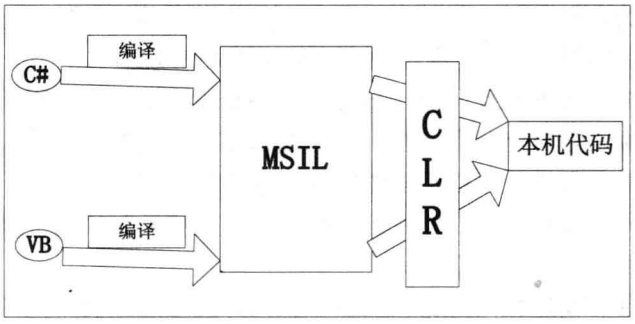


图 1-4　.NET 语言编译过程

当编译为托管代码时，编译器将源代码翻译为 Microsoft 中间语言（MSIL），这是一组可以有效地转换为本机代码且独立于 CPU 的指令。MSIL 包括用于加载、存储和初始化对象以及对对象调用方法的指令，还包括用于算术和逻辑运算、控制流、直接内存访问、异常处理和其他操作的指令。在可以执行代码前，必须将 MSIL 转换为 CPU 特定的代码，这通常是通过实时（JIT）编译器完成的。由于公共语言运行库为它支持的每种计算机结构都提供了一种或多种 JIT 编译器，因此可以在任何受支持的结构上对同一组 MSIL 进行 JIT 编译和执行。

当编译器产生 MSIL 时，它也产生元数据。元数据描述代码中的类型，包括每种类型的定义、每种类型的成员的签名、代码引用的成员和运行库在执行时使用的其他数据。MSIL 和元数据包含在一个可移植可执行（PE）文件中，此文件基于并扩展过去用于可执行内容的已公布的 Microsoft PE 和公共对象文件格式（COFF）。这种文件格式支持 MSIL 或本机代码以及元数据，使得操作系统能够识别公共语言运行库图像。文件中的元数据以及 MSIL 的存在使代码能够描述自身，这意味着不再需要类型库或接口定义语言（IDL）。运行库在执行过程中根据需要从该文件中查找并提取元数据。

在可以执行 Microsoft 中间语言（MSIL）之前，它必须由.NET 框架实时（JIT）编译器转换为本机代码。本机代码是运行于 JIT 编译器所在的同一计算机结构上的 CPU 特定的代码。由于公共语言运行库为每种受支持的 CPU 结构都提供了 JIT 编译器，开发人员可以编写一组可在不同结构的计算机上进行 JIT 编译和执行的 MSIL。然而，如果托管代码调用平台特定的、本机 API 或平台特定的类库，则它只能运行于特定的操作系统上。

JIT 编译考虑了在执行过程中某些代码可能永远不会被调用的事实。它不是花时间和内

存将可移植可执行（PE）文件中的所有 MSIL 转换为本机代码，而是在执行期间根据需要转换 MSIL 并存储结果，本机代码供后面的调用使用。当加载类型时，加载器创建 stub 并将其附加到类型的每个方法。当对方法进行初始调用时，stub 将控制传递给 JIT 编译器，而编译器将该方法的 MSIL 转换为本机代码并修改 stub 以直接执行到本机代码的位置。后面对 JIT 编译的方法的调用将直接进行到以前生成的本机代码，从而减少了实时编译和执行代码所需的时间。

运行库提供了另外一种称为安装时代码生成的编译模式。安装时代码生成模式像常规 JIT 编译器一样将 MSIL 转换为本机代码，但它会同时转换较大的代码单元，并存储结果。本机代码以供随后加载和执行程序集时使用。当使用安装时代码生成时，所安装的整个 assembly 都将转换为本机代码，并且会考虑有关其他已安装程序集的已知信息。与用标准 JIT 选项转换为本机代码相比，结果文件的加载和启动速度更快。

作为将 MSIL 编译为本机代码的一部分，代码必须通过验证过程，除非管理员已经建立了允许代码跳过验证的安全策略。验证检查 MSIL 和元数据以查看代码是否可确定为类型安全，而这意味着它只能访问被授权访问的内存位置。类型安全对于确保对象安全地彼此隔离并由此避免无心或恶意的损坏是必要的。它还提供了对代码可以可靠地强制安全限制的保证。

验证过程中检查 MSIL 代码，尝试确认该代码只能通过正确定义的类型访问内存位置和调用方法。例如，代码不允许以超出内存范围的方式来访问对象。另外，验证检查代码以确定 MSIL 是否是正确生成的，这是因为不正确的 MSIL 会导致违反类型安全规则。验证过程通过正确定义的类型安全代码集，并且它只通过类型安全的代码。然而，由于验证过程的限制，某些类型安全代码可能不能通过验证，而某些语言在设计上就不产生可验证的类型安全代码。如果安全策略要求类型安全代码而代码不能通过验证，则在执行该代码时将引发异常。

1.2 C#简介

C#是微软为.NET 平台设计的程序设计语言，C#拥有 C++的强大特性以及 Visual Basic 简易的特性。

C#是 C/C++家族的第 1 个面向组件和面向对象的语言，与 C++比较，不同点在于完全支持组件的开发模式。C＃可以用来直接开发 ASP.NET 应用程序，因此 C＃的开发者可以直接成为 ASP.NET 的开发者。目前有公司准备将 C＃移植到其他的平台上，并且提供 C#编译器，不久以后 C＃就会成为跨平台的程序语言。

为了使开发者快速进入.NET 框架开发平台，微软推出了 Visual Studio.NET 开发工具，该工具包括 C＃、Visual Basic.NET 与 C++等几种程序语言，借助这个强大的开发工具，开发者可以快速编写.NET 语言程序代码。本书将以 Visual Studio.NET 为工具，介绍 C＃语言。本节主要介绍 C＃应用程序的基本结构。

1.2.1 C#编程环境

由图 1-4 可知 C＃源程序首先需要编译成 MSIL（微软中间语言，以下简称 MSIL），然