

Visual C#

高级编程范例

谭桂华 魏亮 等编著

- ▲ 文件管理器
- ▲ 电子书店
- ▲ 电子银行
- ▲ 网上选课系统
- ▲ 网络聊天工具
- ▲ 分布式会话系统
- ▲ 移动通信管理与收费系统
- ▲ 网络连锁店的集成



清华大学出版社

Visual C#高级编程范例

谭桂华 魏亮 等编著



清华大学出版社

北 京

内 容 简 介

本书以实例详解的方式,循序渐进地介绍了使用 Visual C# .NET 开发应用程序的方法和技巧。全书内容以 C#语言为基础,选取具体典型实例进行讲解。全书共分 11 章,第 1 章简单介绍了 .NET 平台 C# 编程的基础知识,从第 2 章到第 11 章,通过 10 个具体案例的设计与实现,详细地介绍了运用 C#语言编程的方法与技巧。

本书选材新颖,内容完整,涵盖了 .NET 平台下 Windows 窗体和 Web 窗体的编程,以及 XML、COM+、.NET Remoting、Web Service 等高级应用,是一本易懂、实用、极具参考价值的实例教程。本书中的每个实例都调试通过,在每一章的“实例演示”一节中可以看到每个实例程序的运行结果。

本书主要面向具有一定 C#语言编程基础、并准备进一步深入学习 C#编程的读者。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

Visual C#高级编程范例/谭桂华等编著. —北京:清华大学出版社, 2004

ISBN 7-302-08499-8

I. V… II. 谭… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2004)第 033603 号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

组稿编辑:孟毅新

文稿编辑:许书明

封面设计:久久度企划

版式设计:康 博

印 刷 者:北京市世界知识印刷厂

装 订 者:三河市新茂装订有限公司

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:31 字数:716 千字

版 次:2004 年 5 月第 1 版 2004 年 5 月第 1 次印刷

书 号:ISBN 7-302-08499-8/TP·6105

印 数:1~4000

定 价:46.00 元

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770175-3103 或(010)62795704。

前 言

微软公司推出新一代的开发工具 VS .NET，标志着以 .NET 为核心技术的网络时代已经到来。.NET 框架是全新的跨语言软件开发平台，顺应了当今软件工业分布式计算、面向组件、企业级应用、软件服务化、以 Web 为中心等趋势。同时，Visual C# .NET 是微软为 .NET 平台量身定做的程序开发语言，是 .NET 系统的开发利器。

C#拥有 C++的强大功能以及 Visual Basic 简易的特性。为帮助现在所有的 Windows 程序员尽快熟悉 .NET 平台，并能够熟练地使用 VS.NET 快速开发基于 .NET 平台的应用程序，作者编写了本书。

本书从实例入手，通过不同的实例应用，从各个方面对 .NET 做了全面的讲述，将 C#语言的特性和 .NET 技术难点融入到具体的实例中，使读者能够更好的利用 C#来开发应用程序。

本书主要面向有一定 Windows 或 Web 编程经验的中、高级程序员，所以本书没有对 C#语言的语法进行详细的描述，而是从实用角度入手，每一章围绕着开发一个实际应用程序所需的各种 .NET 关键知识而展开，对大家感兴趣的、也是开发 .NET 应用程序中的常用的技术，例如文件系统的访问、数据库操作、COM+应用、远程处理、Socket 编程、电子邮箱、Web 服务的实现等做了详细的讲解。

本书使用了大量示例代码以便读者理解和实际操作，特别对 .NET 中的关键技术进行了非常详细的讲解，对于有一定 C#编程经验而急切地想掌握其高级技术的读者来说，本书无疑是一本理想的学习教材和参考用书，建议将本书与 C#语言的基础教材结合使用，以使效果更佳。

本书共分为 11 章。为了方便读者阅读本书，这里先简单介绍一下本书的组织结构和主要内容。

第 1 章是 C#概述，主要介绍 .NET 框架、基本类库及 C#语法等基础知识。

第 2 章是文件管理器的实现，通过文件管理器这个实例，模拟了 Windows 资源管理器的基本功能。

第 3 章是电子书店的实现，通过 ASP .NET 的强大功能来模拟一个电子书店的简单实现，以此来讲述 ASP .NET 中的用户控件动态加载及 Cookie、应用程序缓存、会话状态等关键技术。

第 4 章是电子银行的实现，通过 COM+技术来模拟实现银行的业务。

第 5 章是网上选课的实现，结合 ADO .NET、ASP .NET 及 COM+应用把网上选课这个实例呈现给大家。

第 6 章是网络聊天工具的实现，主要利用 Socket 来实现一个简单的在线聊天工具。

第 7 章是电子邮箱的实现，通过介绍 POP3 协议和 SMTP 协议来阐述如何构造自己的

电子邮件系统。

第 8 章是分布式会话的实现，通过 .NET Remoting 框架设计了分布式会话的实例，从而实现信息广播的功能。

第 9 章是移动通信的实现，通过移动通信收费功能的设计，详细阐述了 Web Service 的实现。

第 10 章是网络连锁店集成的实现，主要阐述连锁店是如何通过使用 XML，来实现子公司和总公司的交互。

第 11 章是综合性绘图的实现，结合 GDI+ 和 .NET Remoting 设计实现了综合性绘图实例。

在本书中，除第 1 章外，其他各章都分为如下几个部分。

- 提出问题：从需求的角度结合相应的实例演示简单介绍了所选实例的功能，并讲述了所涉及到的关键知识。
- 设计方案：对每个应用实例本身进行详细的设计，具体对应用程序所涉及到的数据库表、实现的功能以及它的层次结构进行了详细的设计。
- 解决方案：根据设计方案给出相应的代码实现。
- 小结：对本章的主要内容、关键技术以及所要注意的问题进行总结。

本书由谭桂华和魏亮执笔编写。蓝荣香、王昊亮、喻波、马天一、魏勇、郝荣福、李光龙、孙明、李天宇、武思宇、牟博超、李彬、付鹏程、高翔、朱丽云、崔凌、张巧玲、李辉、李欣、柏宇、郭强、金春范、程梅、黄霆、钟华、高海峰、王建胜、张浩、刘湘和邵蕴秋等同志在整理材料方面给予了编者很大的帮助，在此，作者对他们表示衷心的感谢。

由于作者水平有限，且创作时间仓促，书中难免有疏漏和不足之处，恳请广大读者批评指正。

作 者

2004 年 4 月

目 录

第 1 章 Visual C#概述	1
1.1 .NET 平台背景	1
1.1.1 第一代 Internet	1
1.1.2 第二代 Internet	2
1.1.3 Microsoft .NET——下一代因特网平台	3
1.2 .NET 框架结构	3
1.2.1 .NET 框架目标	4
1.2.2 公共语言运行时	5
1.2.3 .NET 框架基本类库	6
1.3 C#简介	7
1.3.1 C#的突出优点	8
1.3.2 类型系统	10
1.3.3 变量	17
1.3.4 表达式	20
1.3.5 语句	21
1.3.6 异常处理	24
1.3.7 属性	25
1.4 小结	27
第 2 章 文件管理器的实现	28
2.1 提出问题	28
2.1.1 实例演示	28
2.1.2 System.IO	30
2.1.3 ListView 控件	31
2.2 设计方案	33
2.2.1 文件管理器主界面及功能的设计	33
2.2.2 文本编辑器的设计	34
2.3 解决方案	36
2.3.1 文件管理器主界面及功能的实现	36
2.3.2 文本编辑器的实现	64
2.4 小结	72

第3章 电子书店的实现	73
3.1 提出问题	73
3.1.1 实例演示	73
3.1.2 动态加载及 Cookie	75
3.1.3 应用程序缓存	77
3.1.4 会话状态	79
3.2 设计方案	80
3.2.1 电子书店数据结构的设计	80
3.2.2 电子书店实例的功能设计	81
3.3 解决方案	84
3.3.1 用户控件——好书推荐表的实现	84
3.3.2 BookOrder 类和 ShoppingCart 类	87
3.3.3 Global.asax 文件	91
3.3.4 主页面 BookShop.aspx 的实现	95
3.3.5 ShowCart.aspx 页面的实现	103
3.4 小结	107
第4章 电子银行的实现	108
4.1 提出问题	108
4.1.1 实例演示	108
4.1.2 COM+的基本知识	112
4.2 设计方案	117
4.2.1 数据库层的设计	117
4.2.2 业务逻辑层的设计	118
4.2.3 ATM 界面的设计	119
4.2.4 银行出纳员操作界面的设计	119
4.3 解决方案	120
4.3.1 数据库层的实现	120
4.3.2 业务逻辑层的实现	122
4.3.3 ATM 界面的实现	138
4.3.4 银行出纳员操作界面的实现	148
4.4 小结	163
第5章 网上选课系统的实现	165
5.1 提出问题	165
5.1.1 实例演示	165
5.1.2 ADO.NET	167

5.2	设计方案	170
5.2.1	数据层的设计	172
5.2.2	COM+应用层的设计	173
5.2.3	Web 应用层的设计	174
5.3	解决方案	174
5.3.1	COM+应用层的实现	174
5.3.2	Web 应用层的实现	180
5.4	小结	201
第 6 章	网络聊天工具的实现	202
6.1	提出问题	202
6.1.1	实例演示	202
6.1.2	TCP 协议通信的流程	204
6.1.3	Socket 编程基础	205
6.2	设计方案	207
6.2.1	服务器端的设计	207
6.2.2	客户端的设计	208
6.3	解决方案	209
6.3.1	服务器端的实现	209
6.3.2	客户端的实现	219
6.4	小结	235
第 7 章	电子邮箱的实现	236
7.1	提出问题	236
7.1.1	实例演示	236
7.1.2	电子邮箱的实现原理	242
7.1.3	POP3 协议	242
7.1.4	SMTP 协议	243
7.1.5	协议命令的解析	245
7.2	设计方案	246
7.2.1	数据库的设计	247
7.2.2	客户端的设计	248
7.2.3	服务器端的设计	249
7.3	解决方案	252
7.3.1	客户端的实现	252
7.3.2	服务器端的实现	283
7.4	小结	299

第 8 章 分布式会话的实现	300
8.1 提出问题	300
8.1.1 实例演示	300
8.1.2 .NET 远程处理的框架	303
8.1.3 远程对象、服务器和客户端	307
8.2 设计方案	309
8.2.1 服务器端的设计	309
8.2.2 客户端的设计	312
8.3 解决方案	313
8.3.1 服务器端的实现	313
8.3.2 客户端的实现	318
8.4 小结	331
第 9 章 移动通信管理与收费系统的实现	332
9.1 提出问题	332
9.1.1 实例演示	332
9.1.2 Web Service 基础知识	337
9.2 设计方案	343
9.2.1 数据库的设计	343
9.2.2 Web 服务的设计	346
9.2.3 客户端的设计	347
9.3 解决方案	349
9.3.1 数据库的实现	349
9.3.2 Web 服务的实现	367
9.3.3 客户端的实现	386
9.4 小结	405
第 10 章 网络连锁店的集成	406
10.1 提出问题	406
10.1.1 实例演示	406
10.1.2 连锁店集成程序的背景	411
10.1.3 读取 XML 文档	412
10.1.4 编写 XML 文档	417
10.2 设计方案	419
10.2.1 数据库的设计	419
10.2.2 总公司程序的设计方案	423
10.2.3 子公司程序的设计方案	424

10.3	解决方案	424
10.3.1	数据库的实现	425
10.3.2	总公司程序的实现	429
10.3.3	子公司程序的实现	443
10.4	小结	455
第 11 章	综合性绘图的实现	456
11.1	提出问题	456
11.1.1	实例演示	456
11.1.2	GDI+的编程基础	458
11.1.3	在 IIS 下集成远程服务器对象	459
11.2	设计方案	459
11.2.1	服务器端的设计	460
11.2.2	客户端的设计	461
11.3	解决方案	461
11.3.1	服务器端的实现	462
11.3.2	客户端的实现	466
11.4	小结	483

第1章 Visual C#概述

2000年6月,微软公司推出了“Microsoft .NET 下一代互联网软件和服务战略”,使.NET 利用以互联网为基础的分布式计算和通信激增的特点,通过先进的软件构架,利用网络协议和格式,来协调众多的智能设备与网站,从而提供更为简单、更为个性化和更为有效的互联网服务。.NET 的最终目的就是让用户在任何地方、任何时间,以及利用任何设备都能访问他们所需要的信息、文件和程序,而用户不需要知道这些东西存在什么地方,甚至不需要知道如何获得等具体细节。他们只需发出请求,然后只管接收就可以了,而所有后台的复杂性是完全屏蔽起来的。本章将概述.NET 框架和 C#语言,以使读者对.NET 框架结构和 C#语言有个初始的认识。

1.1 .NET 平台背景

Internet 自从诞生开始,便以迅猛的速度发展、变化着,而应用于 Internet 的技术(如分布式计算)更是推陈出新,呈现百家争鸣的现象。让我们结合 Internet 的发展,先来看看微软为顺应 Internet 的发展所推出的一系列相关的技术。

1.1.1 第一代 Internet

因特网自出现以来,先是以静态的页面做展现,将数据或文件编写成 HTML 页面并放置在因特网上共享,这是所谓的第一代因特网,如图 1-1 所示。

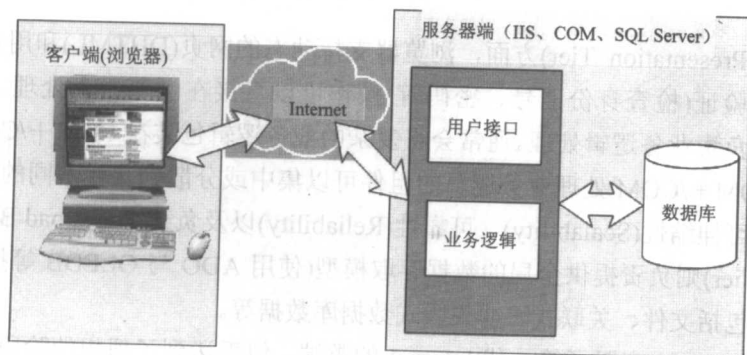


图 1-1 第一代 Internet

从图 1-1 可以看出, UI 逻辑、业务逻辑与数据都存放在服务器端。应用系统在浏览器

与 Web 服务的帮助下, 以类似客户端/服务器(Client/Server)的模式运作。用户从客户端发出一个 HTTP 请求, 服务器端处理用户请求, 产生静态的 HTML 页面, 然后在客户端显示结果。在第一代的 Internet 中, 微软提供了 IE、IIS(Internet Information Services)、COM 等技术来设计网站应用程序。

第一代 Internet 在服务器端实现了数据层(Data Tier)和表现层(Presentation Tier)的分离, 数据归数据, 表现归表现。将应用程序的数据层单独放置在后台的数据库服务器中的优点在于真正达到多人数据共享的目的。然而这种架构却有一定的问题, 首先用户 UI 逻辑和业务逻辑都集中在表现层, 这样使得业务逻辑与接口绑在一起, 虽然这样的做法使得数据库可以随时更换, 但是一旦业务逻辑有所变更, 那么不得不重新编译整个接口程序; 其次, 这种构架不具有可伸缩性(Scalability), 它能够支持的客户端的人数是有一定限度的, 当客户端用户的数目增大时, 这种构架的执行效率便开始下降, 因此便有了新一代的 Internet 的出现。

1.1.2 第二代 Internet

第二代 Internet 是属于 Microsoft DNA(Distributed Internet Architecture)三层式的应用程序结构。应用程序包括三层, 即表现层(Presentation Tier)、业务逻辑层(Business Tier)和数据层(Data Tier), 它的结构如图 1-2 所示。

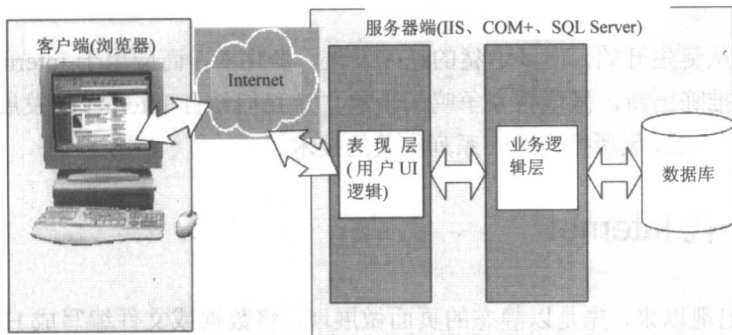


图 1-2 第二代 Internet

在表现层(Presentation Tier)方面, 浏览器支持动态的网页(DHTML)和用户接口的处理逻辑, 如身份验证(检查身份证号、密码等)操作可以直接在浏览器上处理。业务逻辑层(Business Tier)负责业务逻辑处理。通常会将复杂的业务逻辑包装在 COM+/COM 的组件当中, 而通过 COM+/COM 处理业务逻辑的组件可以集中或分散到几台不同的机器上, 利用丛集技术达到可伸缩性(Scalability)、可靠性(Reliability)以及负载均衡(Load Balance)。数据存取层(Data Tier)则负责提供全局的数据存取模型(使用 ADO 与 OLEDB 等技术), 以存取各类型数据, 包括文件、关联式、非关联式数据库数据等。

第二代的 Internet 克服了第一代 Internet 的弊端, 似乎达到了理想的阶段, 但是随着用户需求的变化, 用户已经不再满足于使用 PC 机的软件, 不再满足于只能从 PC 机上网。面对这样的趋势, 微软提出了一个全新的构架——Microsoft .NET。

1.1.3 Microsoft .NET——下一代因特网平台

下一代 Internet 将会是 Programmable Web, 各个网站与各个应用系统之间能够更紧密地整合在一起工作。而 Microsoft .NET 要揭示的是整个世界的因特网即是一个操作系统, 用户在使用因特网上的软件组件时就好像在使用自己公司内部软件组件一样。.NET 对于开发者而言, 使用因特网上的资源就是那么的自然、方便, 用户可以把因特网上的资源直接作为自己的程序的一部分, 而让用户几乎感觉不到它的存在。因此要开发 Information Agent 这类程序, 将不再是一件困难的事情, 并且 B2B、供应链、Application Service Provider 也将真正地落实。实际上不仅如此, 这些 Web Service 更可以在各种不同的 Smart Device, 如手机、PDA、信息家电上执行, 真正达到信息无所不在。下一代 Internet 的结构如图 1-3 所示。

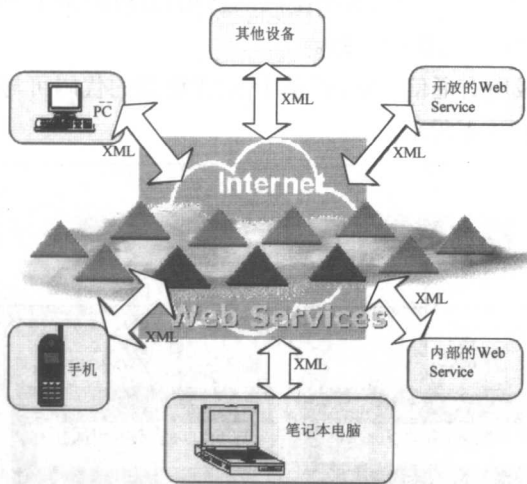


图 1-3 下一代的 Internet

1.2 .NET 框架结构

.NET Framework 是 Microsoft 为开发应用程序创建的一个富有革命性的新平台。从分布式应用系统的角度看, 一个 Web 项目应该具有如下的特点: 能够与用户进行交互; 处理特定的业务功能; 在存储介质中保存它的状态。因此, 这个项目程序应该至少分为三层: 表现层 (Presentation Layer)、事务逻辑层 (Business Layer) 和数据服务层 (Data Layer), 或者分为更多层。

.NET 框架是一种新的计算平台, 它包含了操作系统上软件开发的所有层, 它提供了任何平台上所曾见过的组件技术、呈现技术和数据技术的最丰富的集成级别, 整个体系结构已经被创建为易于在高分布式 Internet 环境中的应用程序开发系统, 就像进行传统的桌面系

统开发一样。

.NET 框架具有两个主要的部件：公共语言运行时和.NET 框架基本类库。下面先看一下.NET 框架的结构，然后我们分别来介绍公共语言运行时(Common Language Runtime)和.NET 框架基本类库。

1.2.1 .NET 框架目标

.NET 框架旨在实现如下目标。

- 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。
- 提供一个一致的面向对象的编程环境，不论对象代码是在本地存储和执行，还是在本地执行还是 Internet 上发布，或者是在远程执行的。
- 提供一个保证代码完全执行的环境。
- 使开发人员的经验在面对类型大小不同的应用程序(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 框架的代码可与任何其他的代码集成。

.NET 框架的结构如图 1-4 所示。

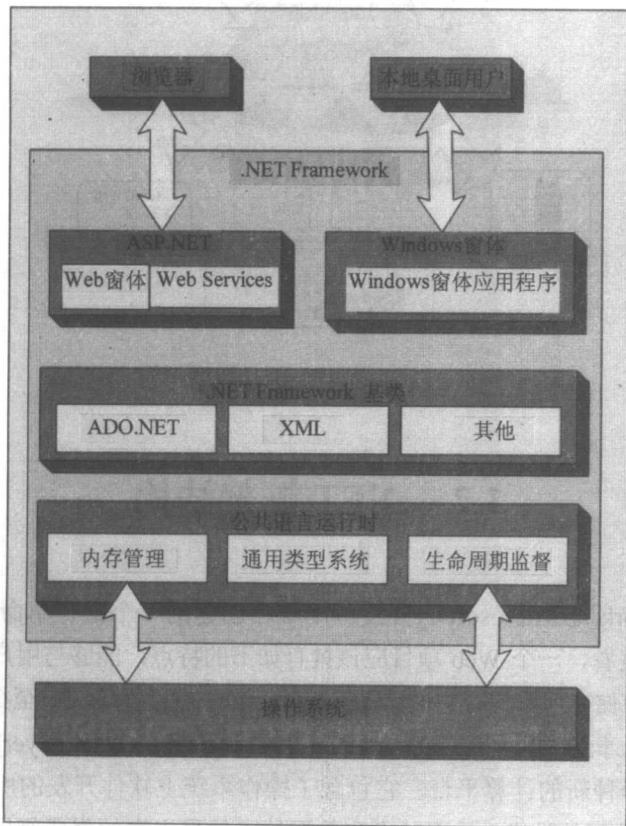


图 1-4 .NET 框架

1.2.2 公共语言运行时

公共语言运行时(Common Language Runtime, CLR)是.NET 框架的基础, 可以将运行时看作一个在执行时托管代码的代理。CLR 提供了核心服务(如内存管理、线程管理和远程处理), 大大简化应用程序的开发, 用户不再需要使用 IDL 来描述组件及接口, 也无须担心忘记调用组件的 Release 函数而造成内存漏失(Memory Leak)的问题, 其结构图如图 1-5 所示。



图 1-5 公共语言运行时的结构

1. 微软中间语言(Microsoft Intermediate Language)

说到虚拟机, 自然会想到 JVM(Jave Virtual Machine)。Jave 跨平台的特性, 其实就是利用虚拟机的原理实现的。所谓虚拟机, 就是仅含指令集的一个通用的虚拟机器。通过虚拟机, 能够将中/高级的语言的程序编译为虚拟机的机器语言程序, 再通过虚拟机的 Runtime(运行时环境), 将这些机器码转化为可以在本地执行的实体机器码。

公共语言运行时 CLR 就相当于虚拟机, 而微软中间语言(Microsoft Intermediate Language, MSIL)就相当于虚拟机的指令集, 它是一种与机器平台、操作系统都无关的虚拟机的语言。无论使用哪一种语言来编写程序, 当使用.NET 的开发工具的编译器来编译程序的时候, 程序将会被编译成 MSIL 的格式。MSIL 告诉 CLR 如何载入和初始化类, 如何调用对象上的方法, 以及如何处理逻辑与算法运算等。在运行时, CLR 的一个组件——即时编译器(Just In Time Compiler, JIT), 将 MSIL 指令集转换为操作系统可以运行的代码。

利用 MSIL 可以实现跨语言的合作, 例如, 可以利用 C#的优越特性设计出程序的核心部分, 再利用 Visual Basic 简便、快速开发的特性, 快速设计出漂亮、美观的程序接口, 供程序的前端使用, 并在 VB 的工程中添加用 C#写的程序核心部分的引用。由于所有的程序都编译成 MSIL 语言, 而所有的语言(C#、VB 等)编译器都能读懂 MSIL 的信息, 因此可以将 C#的工程和 VB 的工程整合, 从而实现了不同语言的整合。Microsoft 已经把基础设置

成允许将 MSIL 代码移植到支持 CLR 的其他的平台。虽然 Visual Studio .NET 和 Windows 2000 提供了运行 CLR 的惟一工具和平台组合,但是可以想像,CLR 是可以移植到其他的平台的,那么 .NET 便真正实现了跨平台、跨语言了。

在 DLL(Dynamic Link Library, 动态链接库)和 MSIL 中都存在着元数据(metadata)。元数据在整个 CLR 中得到广泛的应用,如果要理解 .NET 框架是如何操作的,那么元数据是要掌握的重要概念。元数据提供了 CLR 进行注册、调试、内存管理和安全所需的关于应用程序的信息。比如,对 COM+组件来说,元数据告诉 CLR 和 COM+运行库诸如应当使用的事务处理级别、对象池的最大和最小的容量等方面的信息。由于 MSIL 在延用 Windows 平台使用的 PE(Portable Executable)的基础上对 PE 进行了扩充,把程序的元数据一并写入到可执行文件中,使得 MSIL 程序具有自我解释的能力,这种做法不但可以减少以往的“DLL 地狱(DLL Hell)”的问题,而且也方便了系统对程序的管理,减少了用户的负担。

2. 类型检查器(Type Checker)

CLR 是一个类型安全(Type-Safe)的执行环境,任何不符合类型安全的强制转换操作发生后,类型检查器除了会自动检查这个问题之外,也会自动地检查未初始化的变量与超过索引定义范围的数组。

3. 异常管理器(Exception Manager)

异常管理器提供结构化的异常错误处理机制。在 .NET 之前,不同的程序语言或开发模式拥有不同的错误处理机制,如 Win32 应用程序均是以回传布尔值来表示函数执行成功或失败,而 COM 则利用 HRESULT 的机制,在 Visual Basic 中则使用 On Error Goto 的方式处理。然而不同的错误处理机制使跨语言的软件开发成为一大困扰,而在 .NET 平台中不管使用的语言,都使用一致的结构化异常处理来处理这些异常错误,这样为开发者提供了巨大的便利。

1.2.3 .NET 框架基本类库

.NET Framework 提供了一套操作系统层级的对象函数库,可供程序语言调用使用。.NET Framework 函数库支持 HTTP、XML、SOAP、XSL、Xpath、WebForm 以及 Web Service。图 1-6 所示为 .NET 框架基本类库。

.NET 框架基本类库具有一个重要的特点是,.NET Framework 类函数库提供了统一的应用程序开发模式,它与程序语言紧密结合,将 Class Library 整合到所有支持 .NET 的程序语言中。这样 .NET Framework 类函数库以一致的方式提供所有的功能,任何的程序语言与开发模式都使用一致的 API。.NET Framework 基本类库以阶层式的命名空间(Namespace)及类(Class)组成,并且提供一致的类型系统,现向对象的特性将会更方便用户的使用。另外,.NET Framework 是面向组件的类函数库,属性(Property)、方法(Method)、事件(Event)

等都是主要的构建单元。

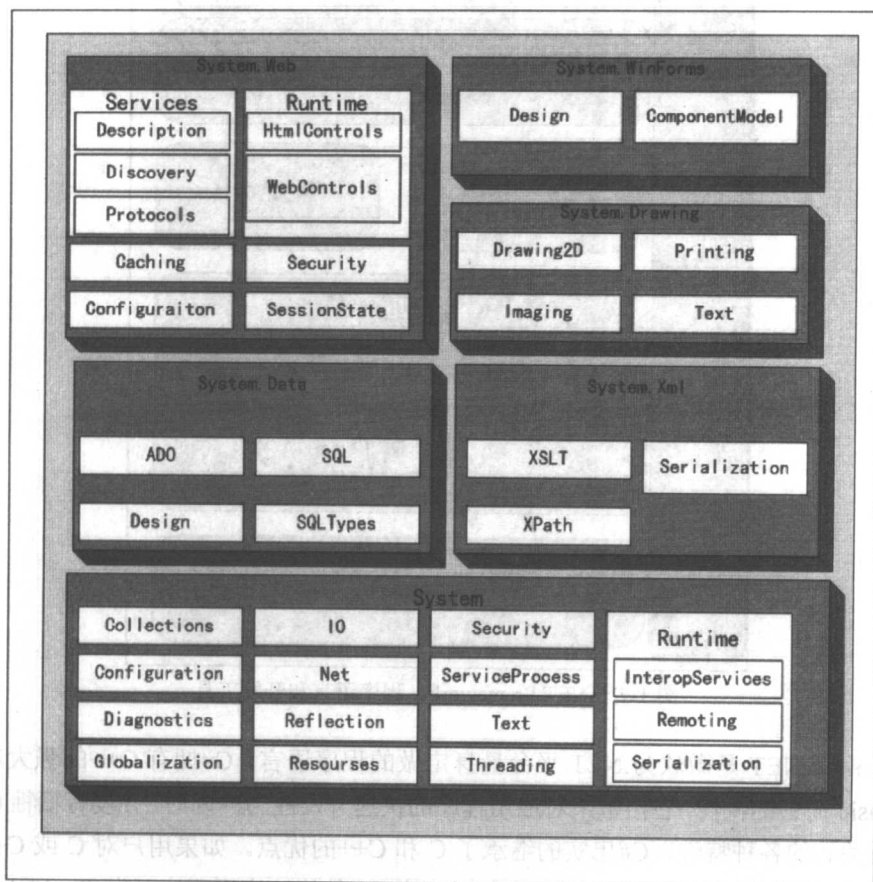


图 1-6 .NET 框架基本类库

1.3 C# 简介

.NET 是一个与程序语言无关的平台，它独立于 .NET 平台之外，所有 .NET 程序语言都可交互运行并访问 .NET 提供的所有服务。

所谓 .NET 程序语言是指符合通用语言规范(Common Language Specification)的程序语言。目前微软已经提供了 Visual C#.NET、Visual Basic.NET、Visual J#、C++ with Managed Extension 等语言。其他厂商也提供对 .NET 的支持，目前已支持的语言包括 COBOL.NET、Pascal.NET、Perl.NET、Smalltalk.NET 等。.NET Framework、程序语言和开发工具的关系如图 1-7 所示。