Delphi 2006 for .NET

开发技术原理与实践教程

蔡 宏 编著
康祥顺 审校

你想成为一名聪明的程序员吗?

你想成为当前最炙手可热的.NET程序设计师吗?

如果你是Delphi的执着追随者，或者是.NET的崇拜者，那么本书将是你直接步入.NET世界的良师益友!

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Delphi 2006 for .NET 开发技术原理与实践教程

蔡 宏 编著
康祥顺 审校

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是一本指导读者如何最大程度地使用Delphi 2006进行Microsoft .NET应用开发的技术实践教程。书中详细介绍了关键的编程概念和利用.NET环境来开发应用程序的基础知识，同时提供了几百个如何在.NET环境下使用Delphi 2006功能的技巧、具有实践性的建议，以及数百个可以立即运行的重要解决方案的详细代码，内容主要涉及以下几个方面：面向对象处理、图形图像应用、ASP.NET应用程序、ADO.NET和BDP .NET数据库应用、Web和Windows窗体、事件和错误处理程序，以及使用多线程技术等。通过本书的学习，读者可以高效地利用Delphi 2006开发.NET应用程序，理解关键操作的内部实现机制，迅速实现大量的编程任务，成为一个当前乃至今后最炙手可热的.NET设计师。

本书适用于大中专院校学生、程序设计人员，以及Delphi和.NET技术爱好者。

图书在版编目（CIP）数据

Delphi 2006 for .NET开发技术原理与实践教程/蔡宏编著.—北京：电子工业出版社，2007.3
ISBN 978-7-121-03825-9

Ⅰ. D… Ⅱ. 蔡… Ⅲ. 软件工具—程序设计—教材 Ⅳ. TP311.56

中国版本图书馆CIP数据核字（2007）第011909号

责任编辑：李 莹
印 刷：北京天竺颖华印刷厂
装 订：三河市金马印装有限公司
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：32.5 字数：820千字
印 次：2007年3月第1次印刷
定 价：48.00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，联系电话：（010）68279077。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

自从Delphi 1.0版发布以来，我就被这款工具所吸引，迄今为止，已经十年有余了。伴随着每次新版本的发布，我总是会在第一时间收到Borland公司的测试盘，随之而来的是欣喜若狂地安装和查看新的特性，久久舍不得放下，因为它给我的工作带来了愉快，极大地简化了设计，提高了开发的效率，这样的工具为何不用呢？我不是真正的程序员，因为我从没有认真地去学习和钻研C++，也没有时间和空间允许我去研究这些真正程序员的技术，但是我足够聪明，所以选择了Delphi。

伴随着微软的Microsoft .NET Framework技术的升温，人们对.NET技术的学习越来越重视。同样，Delphi也发现了软件技术的革命性潮流和趋势，率先发布了支持.NET开发的Delphi 8.0——纯粹的支持.NET开发的平台，随后又发布了融合Delphi Win32、Delphi for .NET、Visual Basic .NET的Delphi 2005，但是这些版本的推出都显得很仓促，不是十分的稳定和成熟，对于Delphi的销售没有什么显著的推动作用。痛定思痛，Borland公司经过努力，开发了集所有支持.NET开发的系统平台Delphi 2006，该版本经过四次测试才最终公诸于众，可以说用心良苦，Borland公司的真正目的是想将这一版本做成当前最为完善的、最为便利的.NET开发平台，让用户不离开Delphi开发环境就能够实现所有的.NET开发，这在当前来说是任何其他的软件平台都没有实现的。

Delphi是Borland公司推出的优秀的前端开发工具。自Delphi问世以来，其友好的集成开发界面、可视化的双向开发模式、良好的数据库应用支持以及高效的程序开发和程序运行，备受广大程序设计师的好评。

近几年，Delphi已成为使用最为广泛的编程语言，而Delphi 2006通过引进结构化错误处理，支持多线程执行，快速建立和使用Web服务的功能，建立新的数据库交互模式（ADO.NET以及BDP.NET）等，又扩大了原有的编程能力。在这之前，要完成这些操作，程序员不得不使用诸如C++或者Java这样的编程语言。

随着Delphi技术的进步和发展，它必将成为今后几年中最佳的编程语言。因此，如果你希望能够在有限的时间和空间范围内完成最为复杂的任务，Delphi将是你的最佳选择，它会给你的工作带来愉悦和效率！

本书按照读者的阅读习惯和程序设计类书籍的学习步骤，组织了程序设计工作中的最为常用的和最容易忽视的知识点，分析了几百个如何在.NET环境下使用各种Delphi 2006功能和特性的技巧。每一个技巧都提供了现成的源代码，可以用来试验某个编程的概念，也可以把它们剪贴到自己的程序中。此外，在介绍每个技巧时还提供了对代码执行过程中的每个步骤的解释。通过阅读本书，你将学习到以下几个方面的知识：

- 学会如何使用Delphi 2006的面向对象编程功能，如类、继承、接口和反射等；
- 学习如何在程序中应用共用语言运行时库和基类库中几千种独立于编程语言的方法和类，以完成特定的任务；
- 学习.NET在支持图形图像和多媒体开发方面的GDI+类库的特性，真正认识到利用GDI+开发多媒体程序的工作是多么简单和快捷；

• 学会在程序中利用丰富多彩的组件构建标准的、友好的、易于操作的用户界面；

• 了解Delphi 2006是如何支持多线程功能的，这一功能可使用户以为你的程序能执行多项任务；

• 领略Delphi 2006如何通过使用Windows窗体，使得基于窗体的用户界面适用于所有编程语言；

• 使用Delphi 2006生成C#语言格式的ASP.NET Web页，通过此项技术的学习，可以让你系统地理解和掌握基于Delphi 2006的ASP.NET Web程序的开发，这也是本书讲述的最为彻底的一部分内容；

• 学会如何使用ADO.NET、BDP.NET取代ADO模式进行数据库访问；

• 一个完整的应用Delphi 2006开发的ASP.NET门户网站，完整地展现了应用.NET技术开发多层结构体系的应用程序的技巧，可以让你在系统学了前面的内容后迅速成为一名专业的.NET设计人员；

• 学会Delphi 2006中的许多操作，以便迅速提高程序的性能和功能；

• 其他更多的内容。

虽然本书尽了最大的努力来展现Delphi 2006开发技术的优势，并且通过实际的案例来详细透彻地讲解相关技术和原理知识，但还是有很多遗憾之处。特别是作为.NET的精华部分——ASP.NET Web Service技术的开发，由于本书篇幅有限，并且这项技术也需要很多的内容才能够详细地讲解清楚，在这里只是简单地介绍了Web Service开发方面的基本知识和应用技巧，如果读者有兴趣的话，可以参考其他相关的介绍Web Service的技术书籍。另外，作为.NET开发的另一法宝——ASP.NET Web应用开发，虽然在本书中用了很多的篇幅来讲解，但是如果要成为一名专业的ASP.NET应用程序开发者，显然这些是不够的，因此，我计划再专门针对Delphi在ASP.NET Web开发方面的技术写一本书（可以参考笔者在清华大学出版社出版的《基于Delphi Web应用开发》一书），这样的话，读者对于Delphi 2006在.NET方面的开发就会学习得比较全面和透彻，也弥补了我和读者的遗憾，同时也能够体现出版社为读者服务的宗旨。

在编写本书的过程中，还有不少的同志付出了辛勤的劳动，他们是黄显堂、刘东鸿、张小丽、李欣、保春艳、杨继锋，在此表示衷心的感谢！另外，电子工业出版社和北京美迪亚电子信息有限公司的全体员工也付出了辛勤的劳动，在此一并表示最诚挚的谢意！

对于本书中的所有实例，编者都在Delphi 2006的开发环境中进行过测试，由于水平有限，尽管做了严格的审核和测试，书中难免仍有一些错误，敬请广大读者不吝赐教！

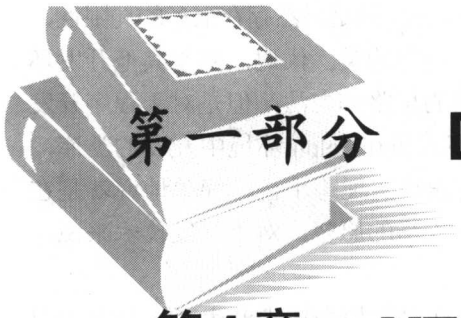
本书所有的源代码均按章节组织，放在了华信教育资源网上，如果读者需要查看源代码，可以到华信教育资源网（http://www.hxedu.com.cn）进行下载。

为方便读者阅读，若需要本书配套资料，请登录“华信教育资源网”（http://www.hxedu.com.cn），在“资源下载”频道的“图书资源”栏目下载。

目　录

第一部分　Delphi for .NET基础

第二部分 Delphi for .NET解决方案



第一部分 Delphi for .NET基础

第1章 .NET技术背景和前景展望

本章将介绍.NET的相关技术背景和前景展望，以及与.NET技术组成相关的核心元素，这对于读者了解一门技术是有好处的，它将决定读者对这门技术学习的信心、前途，以及是否决定使用这门技术来完成自己的工作。但是，如果读者已经对.NET技术有了深入的了解（我指的是这门技术的相关背景知识，而不是它的核心开发技术），决定直接学习它的核心开发技术并且对此抱有信心，那么可以不用看这一章的内容，而直接学习第2章的内容。

本章包括如下知识要点：

- Microsoft .NET技术背景
- .NET是什么
- .NET的优势和前景
- .NET的核心组件——基类库和通用语言运行时库
- .NET应用程序的类型
- 通用类型系统
- Delphi在.NET方面的举措

1.1 Microsoft .NET技术背景

了解或者关注现代信息技术的读者对.NET这个术语并不陌生，在几年前，这个术语在信息技术的营销宣传中使用的频率非常高。但是为什么会冒出这么一项技术并得到业界广泛的宣传和热捧呢？这还得从这项技术的产生背景中深究其原因。

从Windows操作系统本身来说，该系统基础上的软件开发基本上基于一个C语言函数集，最初由3个16位的函数库组成（包括Kernel、User、GDI），后来这3个函数库被一个比较大的32位API函数库所取代。虽然API函数库最初也被看做一个面向对象的体系结构，但是它仍然应用了调用函数库的平面式开发操作。这项技术的应用是十分困难的，对于推广Microsoft的开发技术不太有利。

在20世纪80年代，面向对象的解决方案得到广泛的应用，所有的解决过程都按照函数来实现，但函数特别依赖于指针，十分不稳定且限制较多。在Microsoft的开发技巧中，已经将面向对象的语言引进了类库，这些类库的引进对于开发软件的工作来说，大大地减少了开发的困难。特别是后来Delphi所发展的RAD快速开发技术和VCL可视化组件开发技术，甚至Microsoft的Visual Basic开发环境，不但简化了程序的开发过程，提高了开发应用程序的效率，而且使得很多的开发工作对于大型系统来说可以重复利用。

这些技术的发展对于应用程序的开发的确具有很大的优势，但是仍然存在很多的局限性，就是程序开发人员在某一种开发环境下所做的工作，比如编写的类、代码，对于其他开发环境的程序人员来说没有任何意义，除非只是学习解决问题的思路和方法，但是对于程序开发人员来说，他们要的是效率以及如何简化工作。比如程序人员在Delphi环境中开发的类库，就无法在Visual C++环境下被重复利用，如果有一项技术能够实现跨平台，简单地说就是在不同的环境下相互协作、相互调用，就能够实现程序开发人员的愿望，对于开发技术来说，也是非常伟大的一项进步。

于是Microsoft的COM（Component Object Model）构件对象模型技术产生了，其中也引入了面向对象的技术，扩展了原先的Object Linking and Embedding（对象嵌入与链接，简称OLE）技术，并提供了许多特性，范围从一个可视化的构件模型ActiveX到一个基于MTS（Microsoft Transaction Server）的企业对象和数据库访问体系。这使得用不同的编程语言所编写的应用程序和对象可以使用COM技术来实现共享和相互调用，还可以使用接口定义语言（Interface Definition Language，IDL）或者类型库来公用一个对象的方法和相关的语言定义（如果读者熟悉Delphi 7的程序开发的话，就不难理解这一技巧，因为在Delphi环境中开发一个COM非常简单和可视）。

但是COM的使用也受到很大的局限。COM是一个二进制的对象模型，程序员使用的是一个已经编译了的构件，而类型库不是强制性的，或者说定义不非常明确，因此，如果系统的实现稍有不同，就会妨碍交叉语言的有效使用。另外，COM所在的每个服务器还使用一种不是非常稳定的引用技术机制来分配和释放对应的内存；COM的使用还依赖于用来标示接口和类的全局统一标识数据，而且系统通常把这个数据存在注册表中。这样，就限制了其他程序使用它的可能。

对于COM的升级版本，DCOM、COM+、MTS都采用了一些技术试图来弥补这些不足，但是问题的根本并未得到解决。相反，在技术层面上，Java等可以有效地克服Microsoft的技术局限性，从而在营销市场方面迫使Microsoft不得不拿出新的解决方案来留住客户，甚至说服其他的用户，例如Java使用人员，转向Microsoft技术。

在这种背景下，Microsoft不得不重新写COM解决方案，在经过一段相当长的持续努力后，重写COM的最终结果变为.NET Framework，这是一个与Microsoft的初衷大相径庭的解决方案。但是正如中国的一句古话，“有心栽花花不成，无心插柳柳成荫”，令Microsoft没有想到的是这项技术却成了其营销的一大技术手段，挽救了Microsoft，也挽救了那些沉迷于Microsoft技术的程序开发人员和系统使用人员。

可以说，.NET并不是Microsoft公司所预期实现的，对于他们来说，初期究竟.NET是什么、能够实现什么样的解决方案，也不是十分清晰，只是这项技术（并不一定是一项技术，因为关于这个话题的争论还没有休止，暂且把它叫做一项技术）在后来逐渐地清晰明朗，并且显示了越来越强大的生命力，逐渐地被20多种开发语言所支持。

了解了.NET技术的背景之后，读者需要了解的就是究竟什么是.NET，.NET到底能够实现什么样的目标，前景如何，它与Microsoft其他的解决方案相比到底有什么优势，生命力有多强，用户是否有选择这门技术的充分理由。请先别着急，接下来，咱们就来谈论这个话题。

1.2 .NET是什么

先来看看Microsoft的声音，Microsoft声称，“.NET是Microsoft对于Web Services（Web服务）提供的解决方案，Web服务是下一代的软件，它们以统一的、个性化的方式连接信息、设备和人。”

当Microsoft开始宣传.NET的时候，关于什么是.NET有过很大的争议，甚至有人说，.NET只不过是Microsoft进行市场营销的伎俩，人们不确定.NET究竟是一项技术、新的框架、操作系统，还是一门编程语言。实际上，.NET是基于Microsoft新技术进行分布式开发的Microsoft解决方案的营销“商标”，这种技术称为.NET Framework，可以用来开发Web服务。

从小的范围来讲，.NET主要是用来开发Web服务的，但是它还有很多的优势和应用。到目前为止，读者似乎对于.NET究竟是什么还没有理解到位，对于.NET的含义还很模糊。我要郑重说明的是，.NET是一种市场宣传性质的语言，而不是理解科技概念时的精确语言。要真正明白.NET，就只能通过了解使用.NET的场合来了解其确切的意义：

- 用来指代.NET Framework，一个主要由类库组成的运行时平台和编程框架；
- 包括Visual Studio.NET或者Borland Develop Studio.NET，一个专业的优化集成开发环境，以便更好地使用Delphi for .NET、Visual Basic.NET、C#.NET等符合.NET Framework规范的编程语言进行工作；
- 用来指代.NET Enterprise Server（企业服务器产品的一个程序集），例如Biztalk Server、Exchange Server、Mobile Information Server、SQL Server等，它们为.NET赋予了市场的商标，同时也带来了价值；
- 用来描述.NET的个性化服务，有时也称做Hailstorm，即一系列创建服务的操作，例如清单、合同、进度信息等，这些服务都可以通过与平台和语言无关的方式来访问。

另外一个不可忽视的信息是，对于编程人员来说，.NET的平台和语言无关性才是我们关心的焦点。解决这一问题的方式就是组成.NET最为重要的两个核心元素：.NET Framework基础类库和通用语言运行时库。通过调用.NET Framework基础类库编写应用程序，然后在执行时通过通用语言运行时库的控制，实现平台和语言的无关性，也就是说，在任何编程环境下，只要符合.NET规范，都可以调用.NET Framework类库，使得编译的程序可以在任何平台上运行，例如AMD处理器和Pentium处理器，一次劳动可以重复使用，而不受环境的限制，这就是.NET带给编程人员的好处。

对于长期从事应用程序开发工作的读者来讲，这项技术是非常鼓舞人心的。但是这项技术的生命力和优势怎么样，带给开发人员的利益有多少？接下来，我们就来看看.NET的前景以及优势。

1.3 .NET的优势和前景

对于.NET整体上的优势，前面已经讲得很清楚了，由于这是一本讲解.NET程序设计的技术书籍，而不是研究.NET产品本身的书籍，所以读者需要了解的是，.NET究竟能给程序开发

人员带来什么样的优势，究竟能够为开发的产品带来什么以及为最终的客户带来什么。

• 开发过程更加容易、效率更高

效率是程序开发人员最为关心的。Microsoft已经清楚地意识到，随着Java、开放源代码以及许多Internet脚本语言的发展，很多客户开始趋向于抵制Microsoft的产品，在若干年后，他们会渐渐地失去太多的客户，在这种情况下，聪明的Microsoft只有提供更加高品质的技术、低成本的IDE和免费的技术支持。否则在平台无关性这项技术的推动下，编程人员不会放弃Java等而再继续支持Microsoft。很显然，Microsoft不会坐视不管，他们提供了高品质的.NET Framework，一组用于.NET程序开发的核心类库。

.NET Framework包含一组丰富的、可扩展的类，它为开发人员提供了丰富而强大的开发工具，这些类涉及范围很广，从最低层的系统功能到高级的用户界面设计，都可以利用现成的组件来实现；从简单的桌面型应用程序开发到最为复杂的大型分布式应用系统开发，都变得非常简单；从最为原始的个人PC电脑到最为现代的手持式设备PDA应用的开发，都变得优雅而高效。

更为重要的是，在增加开发效率、近似傻瓜式编程的同时，还提供了高度的灵活性，因为.NET Framework类库不仅仅只包含机械调用的类，这些类是可以扩展的，任何符合.NET规范的语言都可以用于为任何设备开发应用程序。目前支持.NET开发的语言已经达到20多种，包括本书介绍的Delphi语言。

而且在.NET开发过程中，我们不必关心系统底层的任务实现，比如内存管理、类型安全性，因此，可以使开发人员将更多的精力集中到任务的实现上来，任何开发团队在任何.NET开发环境下都可以协同工作，相互调用其他人的代码和对象，这大大地提高了开发的效率。

• 程序的部署更加简单、对系统的依赖性更小

任何开发过应用程序的读者都可能遇到过“DLL地狱”这个词，它是关于Windows以及在上面运行的应用程序和DLL动态链接库的：当用户安装一个新的应用程序时，如果这个应用程序改写了其他应用程序正在使用的DLL链接库，包括版本、时间等，都会使得调用改变了的DLL链接库的应用程序出现问题。另外，在分发应用程序时，都必须利用注册表来维护系统运行，这样，当多次安装和卸载应用程序后，系统将变得非常的“脏”，有时会出现系统运行缓慢和不稳定，在这种情况下，“绿色软件”，也就是面安装的软件受到了用户的欢迎。

.NET已经解决了这个问题，它的安装变得极为简单，只需要将用程序通过复制的方式移动到客户目标机器上即可，甚至在使用一些第三方的COM组件的时候，.NET也可以将该COM对象的库文件自动地复制到目标文件夹，而不需要客户系统上一定安装这些COM。

因此，对于.NET开发人员来说，制作安装程序就没有必要，也许有的读者在浏览Delphi 2006时，很惊讶为什么缺少了专门制作.NET安装文件的工具。

• 客户对于应用系统的访问不受时间和空间的限制

目前，有人持这样一种观点，将来的软件都可以通过租用的方式来获得，用户，包括一些小型的开发商，将不再需要单独开发某个应用程序，他们只需要了解客户的愿望是什么，正如我们使用的电视机一样，无需知道电视机内部究竟怎么运作，而只需要知道怎么开、关电视及一些外在的功能如何使用即可。这个想法在.NET来到后已经变得现实了，这就是Web

服务的理念。

Web服务把整个网络、设备和人想象成一个庞大的计算系统，任何人、设备在任何时间和地点都可以访问他人编写的服务程序，而关于这些程序怎么实现的细节就像被一个黑匣子包装起来了，用户只需要知道怎么使用程序即可。在.NET的解决方案中，编写Web服务程序变得十分简单，当发布服务后，任何需要实现相关功能的客户只需将精力集中到怎么协调实现用户需求等工作上来，而不用考虑怎么实现一个具体的工作。很令人激动的是，这些Web服务程序可以通过任何语言编写，也可以通过任何系统调用。

在Delphi for .NET中，编写Web服务要简单得多，而不像Delphi 7.0以前的解决方案需要编写接口单元和实现单元，它只需要编写服务函数即可。关于这项技术的应用，本书在稍后的章节中将用很大的篇幅来介绍，通过这些章节的学习和实践，读者将会利用所学的知识编写适用、可靠而且具有市场效益的Web服务程序。

·应用程序的跨平台协作更好，可移植性更好

跨平台协作是编程人员最为关心的问题，也是时下各种技术厂家争相宣传和竞争的焦点。对于.NET来说，我们已经说过，通过Web服务可以实现不受时间和空间限制的跨平台协作。但是这是Web服务理念的想法，在现实的程序设计方案中还有一些局限性。使用过Delphi 7.0开发工具的读者可能有过这种境遇，当使用利用.NET开发的Web服务时，在Delphi 7.0环境下，必须使用Soapkit COM服务器来激活它，而不是直接通过创建代理服务器来访问。

但是这种情况在.NET的开发环境中将不复存在，因为.NET Framework的类库本身就可以被任何支持.NET开发的编程语言所使用，因此在程序实现的方式上是一致的，这样的程序在跨平台协作上将变得更加平滑，更加有效。

另外，.NET的目标就是将应用程序应用于各种系统和平台，并且移植到任何设备上，比如PDA、电话等。

·更加安全、可靠

由于.NET程序人员可以验证代码的安全性，所以他们能够获得经过精心调整的代码访问安全性。也就是说，程序员可以决定谁被允许执行某一段指定的代码，这对于用户来说，他们可能遇到很少的病毒的攻击，这将会给目前陷入安全信任危机的Windows系统带来强心剂和安慰，并使之重新赢得用户的支持。但是需要说明的是，.NET是Microsoft用来解决自身问题的方案，目的是推销自己的产品。对于其他的系统来说，到目前为止还没有完全实现，相信在不久会有好消息。

1.4 .NET的核心组件——基类库和通用语言运行时库

前面说过，.NET的核心由两个基本元素支持着：基类库和通用语言运行时库。要学习.NET，就必须首先了解其核心要素，先来看看基类库和通用语言运行时库在.NET体系结构中的层次关系（如图1.1所示）。

从这个层次关系图可以看出编写.NET应用程序的基本过程：首先选择一种.NET编程语言，在编程环境中编写应用程序，这些应用程序通过调用基类库或者扩展的第三方生成的符合通用语言规范的类库，然后在程序运行时通过通用语言运行时库对程序进行控制。为了能

够让应用程序起作用，必须首先选择实现工具，然后考虑符合.NET的规则，接着调用基本的现成的控件、类、代码，最后在执行程序时调用通用语言运行时库。简单地说，.NET基类库用于开发应用程序，通用语言运行时库用于管理应用程序的执行。

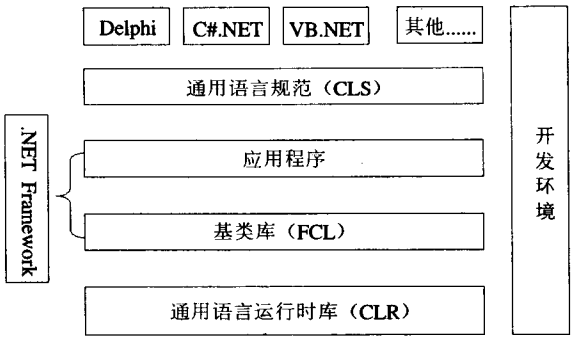


图1.1 .NET各元素的层次关系

从本书讨论的范围来讲，我们需要关心的是基类库和通用语言运行时库。

.NET Framework的基类库

.NET Framework的基类库是类型、类以及接口的庞大程序集，这些类型、类以及接口组成了应用程序的基础，构成了应用程序的蓝图，它能用于创建一些应用程序，包括Windows Form窗体、Web表单以及Web服务应用程序。.NET Framework类型是创建.NET应用程序的基础，它能够执行多项功能，相当于Delphi的VCL库以及Visual C++的MFC通用类型库。

不要以为这没什么不同，事实上，这些类库集成了许多的解决方案，涉及简单的计算处理、网络处理、文件I/O处理、Web处理、异常、线程、队列、数据库等所有应用级别。以作者亲身的体验，编写.NET应用程序相当轻松和直观，可以说是一种享受，相当于用最美味的材料做出色香味俱全的佳肴。特别是当我利用.NET编写WinSock网络处理程序时，我简直不敢相信，原来需要几千行代码才能完成的任务在.NET中仅仅需要调用几个类库就可以完成，这让我只用几分钟的时间就能编完一个简单的局域网聊天程序，或者一个UDP模式的点对点文件处理程序。

通用语言运行时库

通用语言运行时库，简称CLR，管理着编译过的.NET应用程序的执行情况，它的角色类似于Sun公司的Java虚拟机JVM，以及Delphi 7.0中的运行时库。CLR是.NET语言的运行时库，它的功能就是执行和管理以任何能够用于.NET平台的语言编写的所有代码。

在编写.NET应用程序时，可以从众多的编程语言中选择一种，例如Delphi。每一种编程语言都有自己的编译器，编译器通常会把程序代码编译为机器代码，但是特定于语言的及时编译器（Just-In-Time，简称JIT）会将任何的.NET编程语言所编写的代码编译为微软中间语言（Microsoft Intermediate Language，MSIL，简称IL）。在代码的执行过程中，CLR会使用另外一种编译器将中间语言编译为特定于某种运行平台的机器代码。也就是说，CLR既可以将中间语言编译为在AMD处理器上执行的代码，也可以将其编译为在Pentium上运行的代码，这就是.NET解决跨平台问题的秘密所在。

此外，CLR还管理程序执行期间的一些其他事务。

类型和版本信息：.NET编译器不仅仅需要编译为中间语言，还要给编译的可执行文件以及DLL动态链接库文件附加版本、类型和其他的一些信息元素，这样，CLR就可以据此找出应用程序在执行期间的相关性。.NET也可以据此不需要注册表信息，有效地避免“DLL地狱”的发生。

垃圾收集：垃圾收集指的是内存管理自动化。在以前的程序设计过程中，内存的分配和管理都需要程序设计人员自行解决，这样就不可避免地造成“内存泄露”的问题发生。在.NET应用程序中，内存的分配和管理不需要程序设计人员插手，它是由CLR在运行期间自动完成的。当一个实例创建后，系统就分配内存地址；当没有明确地释放内存相当长一段时间后，CLR就会自动地释放内存。但是需要明确，这个自动释放过程是在何时发生的我们无法判断。

代码检查：代码检查是在程序执行前发生的，用来保证程序运行的安全性，并能够确保不会执行诸如被零整除或者一个无效的内存区域这样的非法操作。如果在程序中存在这样的非法代码，CLR将会停止程序的执行，并弹出一个程序异常处理消息对话框。

代码访问的安全性：代码访问的安全性就是根据不同的情况设置不同的访问权限。例如在本机上的应用程序拥有访问本机注册表和文件系统的权限，但是处在内部网络或者Internet网络上的应用程序如果希望进行同样的操作，CLR将会阻止这种请求的执行。

1.5 .NET应用程序的类型

.NET可以创建很多种类的应用程序。

· Web应用程序

Web应用程序在网络发达的今天始终是用户的选择，在最初用HTML语言编写Web应用页面时，完全依赖于设计人员的能力，而且更多的是提供一种静态的页面。但是随着人们对于页面交互性的Web应用程序需求的增加，编写可交互的、动态的Web应用程序就成为Web设计人员的追求。幸好，在.NET中，可以基于可视化的开发技术来开发交互的动态Web应用程序，而且开发的习惯与开发普通的Windows窗体应用程序一样。利用组件的拖放以及对象的属性、事件、方法，就可以快速地开发出专业级的Web应用程序来，这就是ASP.NET Web程序的开发。

使用过Intraweb开发Web应用程序的读者可能还记得可视化所带来的快乐，而在.NET中可以达到同样的高度，且以更加通俗的Web页面的形式存在，控制也更加灵活。

· Windows窗体应用程序

Windows窗体应用程序是所有开发人员的首要选择，当需要丰富的用户界面时，你可以使用.NET提供的强大的桌面应用程序的开发功能。更为安全的是，应用程序可以将数据层、业务规则层和用户表示层分离开来。

在本书大部分的实例开发过程中，基本上使用了Windows窗体类型的应用程序，读者可以发现在.NET中开发该类型的应用程序是多么方便和快捷，开发的程序界面是多么美妙。

· Web服务应用程序

到现在为止，本书已经在很多地方提到Web服务这一概念，这也是.NET宣称的实现目标。