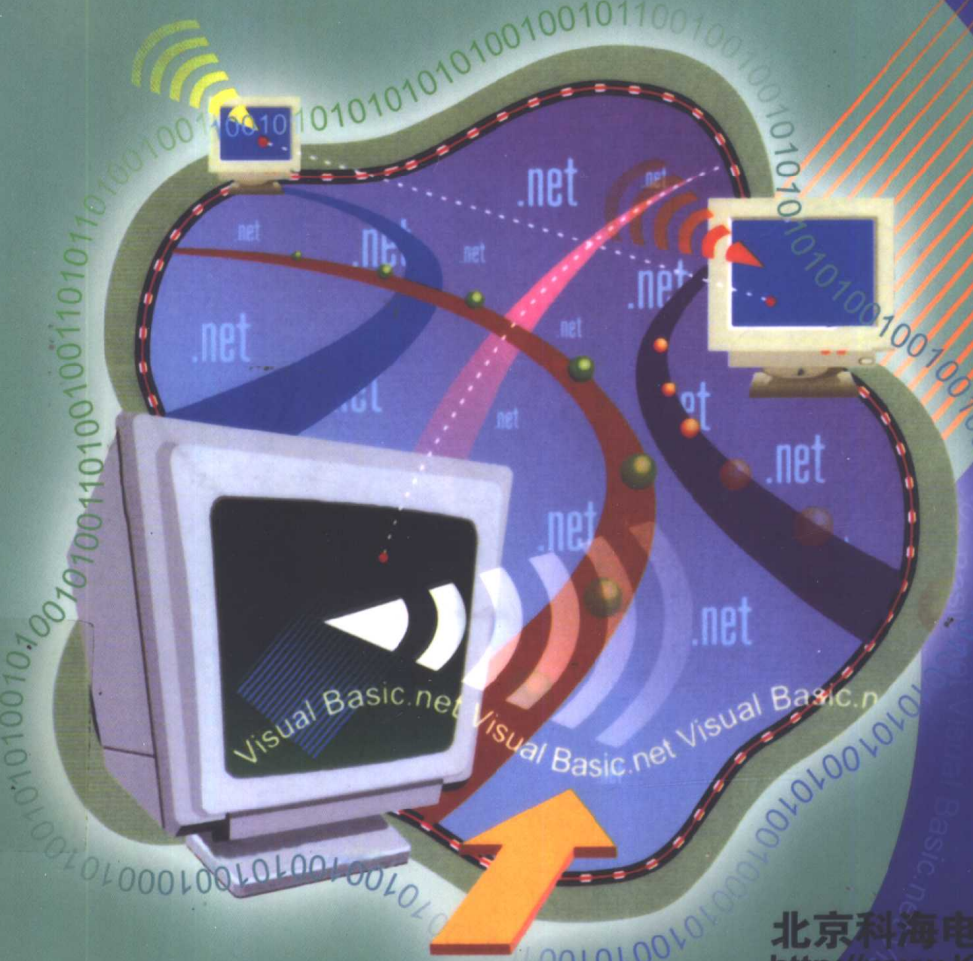
超值光盘

Visual Basic.NET

程序设计示例导学

普悠玛 编著



北京科海电子出版社
http://www.KHP.com.cn

Visual Basic.NET
程序设计示例导学

普悠玛 编著

北京科海电子出版社
http://www.KHP.com.cn

内 容 提 要

本书通过案例全面地介绍了 Visual Basic.NET 程序设计。全书3部分8大章条理清晰、通俗易懂地讲解Visual Basic.NET程序设计中的各个知识点。第一部分简要介绍.NET 运行环境的种种优势以及选择 Visual Basic.NET 的众多因素；第二部分应用“控制台应用程序”逐步深入 Visual Basic.NET 程序的编写，具有侧重点地介绍基本语法、面向对象、委托与事件处理、网络程序、多线程等内容；第三部分应用“Windows 应用程序”进行 GDI+ 图形的绘制与实用程序的开发，了解 Windows Form 的各种属性、事件的设置以及各种控件的使用。

本书是一本易于被初中级学习者接受的Visual Basic.NET程序设计教材，也是对 Visual Basic 有所了解的读者掌握.NET 编程技术的参考资料。

盘　　名：Visual Basic.NET 程序设计示例导学
作　　者：普悠玛
责任编辑：杨雪良
排　　版：杨雪良
光盘制作：王超辉
咨询电话：(010)82896445-8407

出　　品：北京科海电子出版社
印 刷 者：北京市朝阳区科普印刷厂
发　　行：新华书店总店北京发行所
开　　本：异16开　　印　张：19.25　　字　数：420千字
版　　次：2003年2月第1版　　2003年2月第1次印刷
印　　数：0001~5000
盘　　号：ISBN 7-900107-60-6
定　　价：29.00元（1CD/配套手册）

序

蜕变后的 Visual Basic.NET

在步入了.NET 的时代后，Visual Basic 正式宣告死亡了吗？容笔者反问您一个问题，当毛毛虫蜕变为蝴蝶，它是死亡了还是重生了？您的答案将决定 Visual Basic 进入.NET 之后的角色与定位。

蜕变后的 Visual Basic.NET 在功能上着实令人欣赏，对于面向对象的完全支持、多线程的使用、.NET Framework 下各种取之不尽的资源，都将使它成为一个成熟的程序语言，程序语言本身的功能还不是主要的目标，重点在于它将来所要建立的理想世界，一个生活更为便利的世界。

您想要了解并学习.NET 程序设计吗？建议您可以试着从本书开始，除了.NET 基础的执行概念、程序语法之外，还将告诉您如何使用.NET Framework 这个功能强大的函数库，届时您会惊讶于之前您所困扰的种种问题都迎刃而解了。

程序员存在的目的是什么？在于解决人们各式各样的问题！解决问题时您必须有个强大的工具来协助您完成这项工作，您不是花时间又去学习一个全新的语言，您是在发掘一件工具的无尽功能，探索 Visual Basic.NET 所必须花费的时间与精力，已由笔者代您承受而撰写成为您手上的这本书，目的在于使您更快速的入手 Visual Basic.NET，而这正是笔者最初也是最终的目标。

本书导读

本书主要分为三部分：

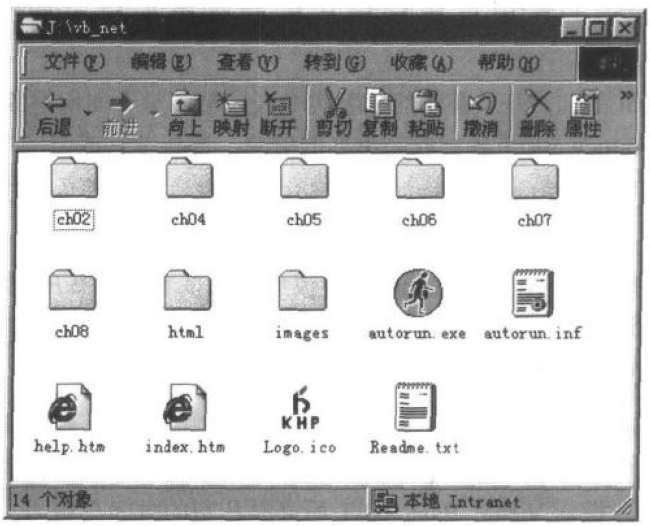
第 1 章～第 3 章是.NET 相关概念的阐述以及 Visual Basic 新、旧版本之间的比较，其目的在于了解整个.NET 构架，并介绍 Visual Basic.NET 在程序开发方面的众多优点。

第 4 章～第 6 章应用“控制台应用程序”逐步深入地介绍 Visual Basic.NET 程序设计，其中有基本语法、面向对象、委托与事件处理、网络程序、多线程等，使你对 Visual Basic.NET 以及.NET Framework 的应用能够充分地认识。

第 7 章～第 8 章详细介绍如何创建 Windows Form 窗口应用程序，说明如何使用各种属性与事件，并且利用 Windows Form 介绍如何使用 GDI+进行绘图。而各种控件的设置与使用，在这部分也有详细的说明。

范例光盘说明

本书中提供有 160 多个范例程序，所有范例程序在书中都标明项目文件名称，您可以在范例光盘中找到相应的范例文件。在执行这些文件前，请按照本书第 2 章介绍的内容安装.NET 运行环境。若要修改范例程序，请将相应的范例程序复制到硬盘中，并去掉文件的只读属性。



普悠玛数位科技

2002/11/15

目 录

第 1 章　认识.NET

1.1　Microsoft 的前景

您正翻开本书的第 1 章，在这一章中可以给您什么？它可以给您一个概念，一个关于什么是 Microsoft.NET 的概念；它可以给您执行的理念，告诉您为什么.NET 的开发工具可以创造微软所描述的未来前景。

不阅读本章或许并不会对以后章节的学习造成任何影响，但是如果您正苦恼于是否要再花时间学习.NET 程序语言开发工具，看完本章的内容，您应该就能有所决定，而笔者也相信您的答案是肯定的。

1.1.1　.NET 与普通用户

就普通用户而言，Microsoft.NET 所带来的是“更为方便的数字生活”以及计算机使用形态的转换。使用计算机的问题已不再是使用何种操作系统、应用哪些软件，而是“使用什么设备”、“什么时候使用”及在“什么地方使用”等问题。

使用什么设备

比尔·盖茨（Bill Gates）的想法之一，是让每一个家庭都拥有个人计算机（PC）。这个梦想可以说已经实现了大半，其下一步计划就是改变每个人对计算机的使用习惯，而第一步就是改变计算机的使用形态，使所谓的计算机不再只是桌上那个四四方方的物体。

想象将来您可以在 PDA 上使用各种软件，利用手机可以满足各种通信需求，冰箱、电视、相机、电话等各式家电，任何您所能想到的电器用品上都会有计算机设备的存在。这就是 Microsoft.NET 所描绘的美好前景，届时可能也就没有“个人计算机”这个名词存在。

什么时候使用

什么时候可以使用到数字服务？只要你想使用，就可以使用。所有的数据都存在于网络中，网络就是个巨大的数据库，全年 24 小时无休，当您想要查找数据或使用某个软件时，使用身边的任何一款设备，都可以启动所需的应用程序或获取所需的信息资料。

什么地方使用

在什么地方可以使用数字信息服务？答案是在任何地方，也就意味着不用再带着计算机到处跑。无论您是在机场、动物园，还是在开车的路上，随时都可以连上网络并使用所需的服务，什么？软件没有安装该怎么办？届时就已经不存在所谓的软件安装问题，所有的软件服务都可以直接租用，而 Setup 或 Install 等名词也将远离我们的记忆。

1.1.2 .NET 与程序开发人员

现在的程序开发人员若不精通 4、5 种语言，可能都无法在程序设计领域生存。当 Microsoft.NET 提出 Visual Basic.NET、C#、J#等语言开发工具时，程序开发人员的第一反应就是“天啊，又要学习一种新的语言了”！此时，您会做出什么决定呢？是接受学习，还是继续固守您目前所擅长的语言？

Microsoft.NET 究竟对程序开发人员有什么样的改变？首先，我们还是看看程序开发人员目前所面临的各种问题。

过多的语言开发工具

现在的程序设计领域存在一个严重的问题——过多的语言开发工具。这并不是一种好现象，每个程序员在学习新语言上所花费的成本过高。而程序语言之间不能互通，使用不同语言开发的程序彼此之间也很难进行沟通与合作（即使是微软之前所使用的 Visual C++ 与 Visual Basic 在沟通上也存在困难）。

在.NET 的框架下，程序员可以使用自己所擅长的语言进行程序的开发，可以使用 Visual Basic.NET、C#、J#或其他支持.NET 的程序语言（所谓支持就是指将原有语言进行些调整，例如数据类型或语法等必须共同遵守的基本规范）。不过，程序员能使用自己所擅长的语言进行开发并不是.NET 的最大优点，重点在于所开发的程序相互之间可以相互沟通、支持。

跨平台的问题

跨平台的问题一直是所有程序开发人员想要突破的瓶颈，由于机器与操作系统的不同，造成所开发的程序只能在单一平台上执行，程序开发人员若要在不同的平台上执行程序，就必须为每个平台撰写专用的程序（就算是 Windows 操作系统，也无法保证程序在各种版本下都可以运作），当然也就造成开发成本的增加。

在.NET 框架下，跨平台的梦想将成为可能，最基本的要求是必须安装.NET 运行环境（Runtime），无论您是在何种平台下进行程序的开发，转到另一个平台马上就可以运行，安装程序只要一个简单的复制操作就可以完成，一次开发就可以适用于多种平台。

可重用组件

程序开发人员在编写程序时，若要自行开发所有的组件，将会相当没有效率。如何善用各种可重用组件加强程序的功能与提高开发的效率，才是最重要的课题。不过，可重用组件的使用却也是一个令程序开发人员苦恼的问题。

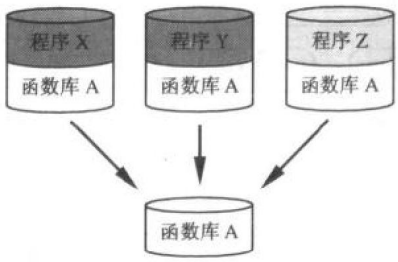
可重用组件包括了前面两个问题，第一就是使用不同语言所开发的组件所引发的沟通问题；另一个就是组件的跨平台问题，使用了 Windows API 的组件势必就无法在其他操作系统下进行运行。另外，组件的版本问题（例如 DLL Hell）也经常发生，所有这些问题在.NET 框架中将获得解决。

.NET Framework 提供了一个巨大的“基础类库”，该类库可以被各种支持.NET 的语

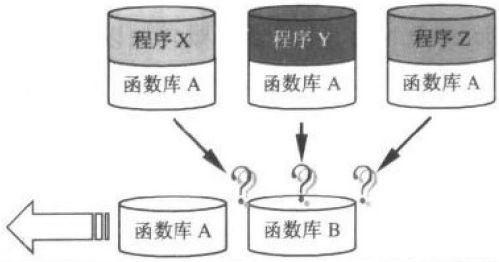
言使用，跨平台的问题也就可以得到解决，而程序员也只要遵守一些基本的规则，就可以避免 DLL Hell 这类的版本问题。

什么是 DLL Hell?

DLL 全名为“Dynamic Linking Library”，其原本目的是用来弥补 Windows API 功能的不足，以及节省内存的使用，程序在编译时并不将所使用到的 DLL 一同编译，而是直接引用，如下图所示。



由于以引用的方式使用 DLL 中的函数，确实是节省了内存的使用，但是如果当 DLL 版本更新了，原先引用旧版本的程序将可能无法使用（虽然大部分的情况下新版本号称可以兼容于旧版本），这就是 DLL Hell，请看下图进行理解。



1.1.3　网络操作系统

若要谈到跨平台的问题，这并不是 Microsoft.NET 的最终目标。在 Microsoft.NET 的设想下，是要将整个网络作为执行应用程序与开发程序的平台，网络中有各种可用的应用程序与开发程序时可用的组件，网络本身就是个巨大的操作系统，这么说似乎有点抽象，以下的例子有助于您对内容的理解。

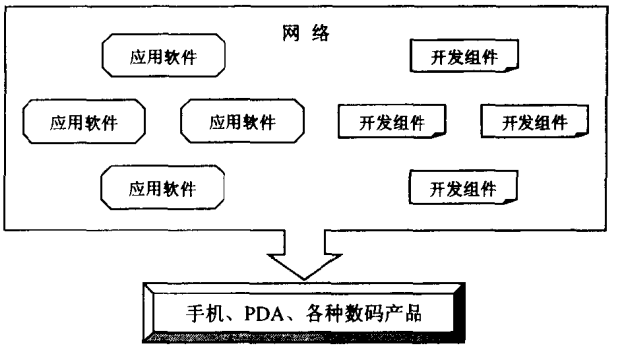
软件租用

当您需要使用某个应用软件时，可以连上网络直接租用该软件，如直接从网络上调用文字处理软件、电子邮件服务、绘图程序等。这似乎有些不可思议，但在.NET 框架下却是可行的，分散在世界各地的网站将提供所需的各种软件。

“软件租用”概念是好的，因为大部分用户对于一项软件或许根本使用不到其功能的三分之一，用户可以通过软件租用使用各自需要的功能。

开发程序

程序开发人员可引用的组件不再局限于本机所提供的组件，可以直接引用网络上所提供的组件服务，世界各地的网站上将提供可以使用的各种函数库。注意，您是从网络直接引用，而不是通过下载文件使用这些组件。

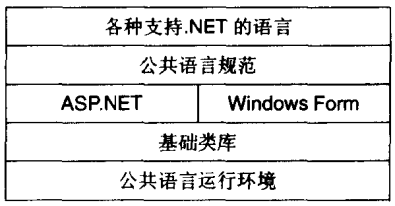


以上概念可以通过 Web Service 实现。Web Service 是以 ASP.NET 为基础，通过 XML 传递数据，以 HTTP 作为通信协议的。假如 Web Service 的设想可以实现，网络就真的是个巨大的操作系统了。关于 Web Service 的运行，在下一小节将有更为详细的介绍。

本小节主要针对 Microsoft.NET 的前景与概念加以介绍，下一小节将开始探讨.NET Framework 的框架与运行方式，看看它怎样实现.NET 所承诺的种种前景。

1.2 .NET Framework 框架与运行

.NET Framework 是微软的新一代程序开发环境，其主要框架结构如下图所示。

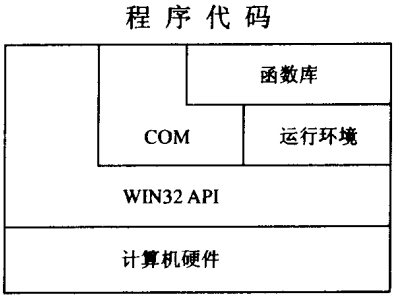


上图是.NET Framework 的大致框架，笔者将由下往上逐步解析各个方块内的细部结构与运行方式。

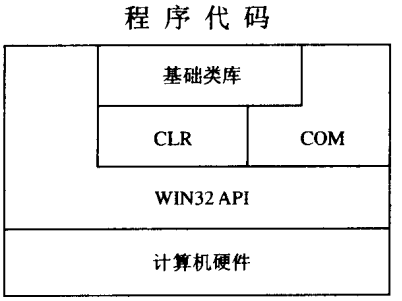
1.2.1 公共语言运行环境

在认识“公共语言运行环境”之前，首先认识一下何谓虚拟机（Virtual Machine，简称

VM）。虚拟机是用来描述开发程序时所使用的平台，并包括了运行环境（Runtime）及如何与硬件进行沟通。每个程序语言有其各自独特的虚拟机，以 Visual Basic 为例，其 6.0 版本的虚拟机结构如下图所示。



以往版本的 Visual Basic 运行速度一向是为人所指责，这是因为其执行环境是以 COM 为基础进行编译与运行的。在.NET 框架下，所有的程序语言将使用统一的虚拟机，公共语言运行环境（Common Language Runtime，简称 CLR）也将是所有的.NET 语言在执行时所必备的执行环境，.NET 虚拟机的结构如下图所示。



统一虚拟机与运行环境是达到跨平台目标的第一步。在 CLR 中，大致可以再分为以下几个框架。

通用类型系统

通用类型系统（Common Type System，简称 CTS）的作用在于使所有的语言共享相同的数据类型，数据类型与名称的不统一是无法达成跨语言的一个原因。CTS 内置于 CLR 中，无论程序是以什么语言编写而成，都会被编译成相同的中间语言，而这个中间语言在 CLR 下时其数据都将具有相同的名称，从而使得不同语言之间的数据得以沟通协调。

内存管理与 Garbage Collection

支持.NET 且愿意遵守一些共同规范的程序语言所编写的程序代码，称之为“managed code”，而之前各版本的其他语言编写的代码称为“Unmanaged code”。被称为“managed code”的程序语言是因为这些程序代码在执行过程中所使用到的内存资源将受到 CLR 的监

控，各种数据与对象的生存期都将由 CLR 管理。

在.NET 的 CLR 中具有“Garbage Collection”的资源回收机制，在 C++中如果对象在清除之前没有将所使用到的资源加以释放，可能会造成程序错误或内存的耗损；在 CLR 中会自动管理所产生的对象，当 Garbage Collection 检测到对象不再被使用与引用时，就会将对象清除并释放其所使用的资源，该操作是自动且无法预测的，您并没有办法获知 Garbage Collection 何时会被启动。

中间语言与实时编译器

在.NET 框架下，所有的程序语言在编译时会先转为与平台机器无关的“中间语言”（Microsoft Intermediate Language，简称 MSIL 或 IL），MSIL 会与 metadata 一同编译成 PE（Portable Executable）可执行文件，metadata 描述了程序代码中所使用到的各种类型与数据，MSIL 与任何的机器平台无关，但相当接近于机器的机器码（machine code）或本机代码（Native code），并可以在安装有 CLR 的机器上执行。

将程序运行时必需的信息写入到同一个文件中有个好处，该程序将有能力“自我描述”（self-describing），而不再需要向操作系统的注册表（Registry）进行登记，安装程序就只需简单的复制操作，也能避免 DLL Hell 的问题发生。

当程序第一次被执行时，CLR 会启动“实时编译器”（Just-In-Time Compiler，简称 JIT）。JIT 是动态的，它会侦测硬件设备而将程序进一步编译为该机器的本机代码，以确保程序在任何一个平台上都能执行。

综合以上所述，笔者将 CLR 的作用整理为如下内容：

- 定义共同的类型
- 管理对象与数据
- 自动执行 Garbage Collection
- 使用 JIT 将 MSIL 编译成机器码

1.2.2 统一的类库

程序开发过程中，会有许多的功能组件被重复使用，于是就将这些组件制作成类库，每一种程序语言都拥有各自独立的类库，如 C++的 MFC 或 ATL，Java 的 JDK 等。然而，每一种类库都是针对一种语言的，所以这些类库彼此之间并不能互相引用。对于偏好 Visual Basic 的程序员而言，所开发的类库就无法被 C++程序员使用。

.NET Framework 下提供了一个巨大的统一的类库（Unified Class Library），该类库提供了程序开发人员在开发程序时所需要的大部分功能，重点是这个类库可以使用任何一种支持.NET 的程序语言加以引用，程序员不再需要为了不同的类库而学习不同的程序语言。

.NET 所提供的类库是以面向对象为基础进行创建的，其实在.NET 框架下，不管是数字还是字符串，所有的数据都是对象。统一的类库结构是阶层式的，采用 Namespace 加以