

魅力·实践·发现

Visual C++ .NET 2010

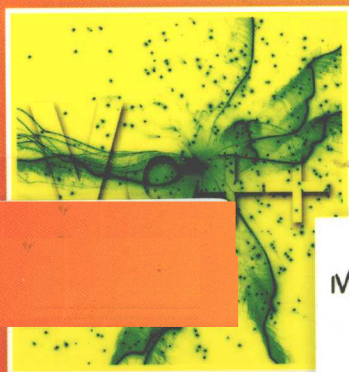
开发实践——基于C++/CLI

郑阿奇 主编

展现精英高手发现之旅

站在流行平台开发实践

介绍流行软件神奇魅力



Microsoft
Visual
Microsoft

VISUAL
Studio

简体中文



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

魅力·实践·发现

Visual C++.NET 2010 开发实践 ——基于 C++/CLI

郑阿奇 主 编

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书首先介绍 C++/CLI 编程和 C++/CLI 面向对象编程基础,然后介绍了在 Visual Studio 2010 (中文版) 平台中,采用 C++/CLI 编写应用程序的方法。内容包括 Windows Forms 编程基础、文件与数据串行化、GDI+ 编程基础、多线程编程技术、网络应用编程和数据库应用编程。本书包括教程、实验和综合实习三个部分。通过教程学习,结合上机实验和实习,读者能够在比较短的时间内掌握 Visual C++.NET (基于 C++/CLI) 应用技术。

本书可作为 Visual C++.NET (基于 C++/CLI) 技术培训和应用开发的参考用书,也可以作为大学本科和高职高专有关课程的教材。

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 C++.NET 2010 开发实践——基于 C++/CLI / 郑阿奇主编. —北京: 电子工业出版社, 2010.12

魅力·实践·发现

ISBN 978-7-121-12153-1

I. ①V… II. ①郑… III. ①C 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2010) 第 212338 号

责任编辑: 郝黎明 特约编辑: 张 慧

印 刷: 涿州市京南印刷厂

装 订: 涿州市桃园装订有限公司

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

开 本: 787×1092 1/16 印张: 25.75 字数: 659.2 千字

印 次: 2010 年 12 月第 1 次印刷

印 数: 4000 册 定价: 45.00 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 zltz@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

随着.NET技术的不断发展,.NET Framework从早期的.NET Framework 1.0发展成为目前的.NET Framework 4.0,并被集成在Visual Studio 2010开发平台中。.NET Framework是Visual Studio 2010开发平台的核心组件,主要由CLR(Common Language Runtime,公共语言运行时)和.NET Framework类库两个核心部分组成。CLR提供了用户程序执行的环境,而.NET Framework类库提供了在CLR中执行程序代码时所需的支持。在编写.NET程序时可以用C++、C#或者任何其他支持.NET Framework的语言编写,而这些语言都使用相同的.NET程序库。虽然Visual Studio 2010仍然支持Visual C++6.0 MFC模式开发的程序,但Visual Studio 2010开发平台推出了全新的C++/CLI语言标准,这个全新的标准是对目前C++标准的扩展,这样既可以继续使用传统C++提供的全部功能,拥有C#般的编程效率,而且还可以使用.NET提供的强大功能。

C++/CLI语言标准比较新,国内介绍C++/CLI语言标准资料比较少。我们在认真研究C++/CLI语言标准的基础上,结合Visual Studio 2010实践,推出本书,旨在系统介绍在Visual Studio 2010平台下,采用C++/CLI语言标准,开发应用系统的方法。

本书首先系统介绍C++/CLI编程基础和C++/CLI面向对象编程基础,然后系统介绍在Visual Studio 2010(中文版)平台下,采用C++/CLI编写应用程序的方法,包括Windows Forms编程基础、文件与数据串行化、GDI+编程基础、多线程编程技术、网络应用编程和数据库应用编程。本书包括实用教程、实验指导和综合实习三个部分。通过教程学习,结合上机实验和实习,使读者能够在比较短的时间内掌握Visual C++.NET(基于C++/CLI)应用技术。

本书由南京师范大学郑阿奇主编,参加本套丛书编写的还有郑进、陶卫冬、邓拼搏、严大牛、韩翠青、王海娇、刘博宇、孙德荣、吴明祥、周何骏、徐斌、孙承龙、陈超、毛风伟等。

本书配有教学课件和应用系统的所有源代码文件,需要者可以从出版社网站免费下载。网站地址为:<http://www.hxedu.com.cn>。

由于水平有限,错误在所难免,敬请广大师生、读者批评指正。

意见建议邮箱: easybooks@163.com

编 者

2010.12

目 录

第一部分 实用教程

第 1 章 C++/CLI 编程基础	1
1.1 .NET 与 C++/CLI	1
1.1.1 .NET 框架概述	1
1.1.2 公共语言运行时	3
1.1.3 C++/CLI 概述	4
1.2 简单的 C++/CLI 程序	4
1.2.1 创建 CLR 控制台程序	4
1.2.2 命名空间	6
1.2.3 应用程序入口	7
1.2.4 控制台输入/输出	8
1.3 基本数据类型	11
1.3.1 基本数据类型	12
1.3.2 装箱与拆箱	13
1.3.3 强制类型转换	14
1.4 C++/CLI 枚举	16
1.4.1 枚举的定义	16
1.4.2 枚举的常量	17
1.4.3 枚举常量的运算	17
1.5 C++/CLI 数组	18
1.5.1 跟踪句柄	18
1.5.2 CLR 数组	19
1.5.3 数组的排序及查找	21
1.5.4 多维数组	25
1.5.5 数组的数组	26
1.6 C++/CLI 字符串	27
1.6.1 字符串的定义	27
1.6.2 字符串的常用操作	28
1.6.3 内部指针	33
1.7 函数及泛型函数	34
1.7.1 函数的定义及调用	34
1.7.2 参数可变函数	35

1.7.3 泛型函数	36
第 2 章 C++/CLI 面向对象编程基础	39
2.1 数值类和引用类	40
2.1.1 数值类	41
2.1.2 引用类	42
2.2 类的成员	44
2.2.1 类和程序集	44
2.2.2 字面值字段与只读字段	47
2.2.3 类的构造函数	48
2.2.4 类的终结器	50
2.3 类的属性	52
2.3.1 标量属性	52
2.3.2 索引属性	54
2.3.3 静态属性	55
2.4 类的继承和多态	55
2.4.1 类的继承	56
2.4.2 接口类	59
2.4.3 类的多态	61
2.5 委托和事件	64
2.5.1 委托的定义	64
2.5.2 无约束委托	66
2.5.3 事件及其处理	68
2.6 通用类及通用集合类	69
2.6.1 通用类的定义	69
2.6.2 通用集合类	72
2.7 异常处理与调试	80
2.7.1 C++/CLI 异常处理	80
2.7.2 常见.NET 异常类	81
2.7.3 Debug 类和 Trace 类	82
2.7.4 使用调试类和跟踪类	83
第 3 章 Windows Forms 编程基础	89
3.1 Windows 窗体基础	89
3.1.1 创建 Windows 窗体项目	89
3.1.2 窗体应用程序的结构	91
3.2 窗体及其属性、事件	93
3.2.1 Form 类	93

3.2.2	窗体属性	93
3.2.3	窗体事件	95
3.3	Windows 通用控件	97
3.3.1	标签	97
3.3.2	按钮	98
3.3.3	文本框	99
3.3.4	图片框	102
3.3.5	列表框	102
3.3.6	组合框	103
3.3.7	进度条	104
3.3.8	定时器	105
3.4	菜单、工具栏及状态栏	107
3.4.1	菜单栏	107
3.4.2	工具栏	111
3.4.3	状态栏	113
3.5	多文档界面 (MDI)	114
3.5.1	多文档界面的结构	114
3.5.2	创建多文档界面窗体	115
3.5.3	合并菜单及工具条	116
3.5.4	管理 MDI 子窗体	120
3.6	对话框及通用对话框	122
3.6.1	模式与无模式对话框	123
3.6.2	通用对话框	124
3.6.3	消息提示对话框	126
第 4 章	文件与数据串行化	128
4.1	文件类及数据流	128
4.1.1	System::IO 命名空间	128
4.1.2	读写数据流	129
4.2	目录的常用操作	130
4.2.1	Directory 类	130
4.2.2	常用目录操作	131
4.2.3	DirectoryInfo 类	134
4.3	文件的常用操作	136
4.3.1	File 和 Path 类	137
4.3.2	常用文件操作	139
4.3.3	FileInfo 类	144

4.3.4	读写文本数据	146
4.3.5	读写二进制数据	149
4.4	数据串行化	153
4.4.1	串行化与反串行化	154
4.4.2	二进制串行化方式	155
第 5 章	GDI+编程基础	160
5.1	GDI+概述	160
5.1.1	GDI+的组成部分	160
5.1.2	GDI+的接口架构	161
5.2	图形绘制	162
5.2.1	画笔 Pen 类	162
5.2.2	画刷 Brush 类	164
5.2.3	绘制二维图形	166
5.2.4	绘图和填充示例	169
5.3	图像处理	171
5.3.1	图像 Image 类	171
5.3.2	加载及显示图像	172
5.3.3	裁切及缩放图像	176
5.3.4	旋转、反转及扭曲图像	177
5.3.5	图形对象变换	178
5.4	文本显示	181
5.4.1	字体 Font 类	181
5.4.2	简单文本输出	182
5.4.3	反锯齿化处理	183
5.4.4	格式文本输出	185
5.5	打印及打印预览	188
5.5.1	PrintDocument 类	188
5.5.2	打印相关的对话框	189
5.5.3	绘制打印页面	189
第 6 章	多线程编程技术	194
6.1	进程与线程	194
6.1.1	进程与线程的关系	194
6.1.2	多线程的工作方式	195
6.2	.NET 对多线程的支持	196
6.2.1	多线程 Thread 类	196
6.2.2	线程的创建与启动	197

6.2.3	线程的状态与优先级	198
6.2.4	线程的挂起、恢复及终止	201
6.3	托管线程池与定时器	203
6.3.1	托管线程池	203
6.3.2	定时器	204
6.4	线程的同步	205
6.4.1	.NET 对同步的支持	205
6.4.2	使用 Monitor 类同步线程	206
6.4.3	使用 Mutex 类同步线程	209
第 7 章	网络应用编程	213
7.1	网络通信概述	213
7.1.1	网络通信模型	213
7.1.2	网络通信协议	215
7.1.3	IP 地址及端口	216
7.1.4	.NET 网络接口	217
7.2	套接字概述	219
7.2.1	套接字 Socket 类	220
7.2.2	面向连接的套接字	222
7.2.3	面向无连接的套接字	225
7.3	同步 TCP 应用编程	225
7.3.1	NetworkStream 对象	226
7.3.2	TcpClient 和 TcpListener 类	227
7.3.3	发送和接收数据	230
7.3.4	TCP 无消息边界处理	234
7.4	异步 TCP 应用编程	235
7.4.1	异步操作方式	235
7.4.2	异步接收连接请求	237
7.4.3	异步发送连接请求	238
7.4.4	异步发送和接收数据	239
7.5	UDP 应用编程	245
7.5.1	UdpClient 类	245
7.5.2	发送和接收数据	246
7.5.3	广播和组播	249
第 8 章	数据库应用编程	255
8.1	ADO.NET 概述	255
8.1.1	ADO.NET 体系结构	255

8.1.2	ADO.NET 对象模型	257
8.2	数据库访问核心类	257
8.2.1	数据库提供者核心类	257
8.2.2	ADO.NET 核心类	259
8.3	创建数据库连接	260
8.3.1	Connection 对象	260
8.3.2	连接字符串	261
8.3.3	连接数据库	262
8.4	连接的数据库操作	263
8.4.1	Command 对象	263
8.4.2	执行 SQL 语句	264
8.4.3	执行存储过程	266
8.4.4	DataReader 对象	268
8.5	断开的数据库操作	271
8.5.1	DataAdapter 对象	271
8.5.2	DataSet 对象及其结构	272
8.5.3	修改 DataSet 中的内容	275
8.6	数据绑定	279
8.6.1	创建非类型化 DataSet	279
8.6.2	数据访问控件	281
8.6.3	控件的数据绑定	285

第二部分 实验指导

实验 0	熟悉 Visual Studio 2010 开发环境	289
实验 1	C++/CLI 编程基础	295
实验 2	面向对象编程	300
实验 3	继承和多态	306
实验 4	窗体及通用控件	311
实验 5	菜单、工具栏和状态栏	317
实验 6	文件及数据串行化	322
实验 7	GDI+图形绘制	327
实验 8	文字及打印	332
实验 9	多线程编程	336
实验 10	网络应用编程	342
实验 11	数据库应用编程	349

第三部分 综合实习

学生成绩管理系统	353
----------------	-----

附 录

附录 A 程序调试	372
附录 B 数据库准备	377
附录 C 安装部署程序	388

第一部分 实用教程

第 1 章

C++/CLI 编程基础

C++/CLI 语言标准是专门为编写依赖 CLR 的应用程序而设计的,该标准是对目前使用的标准 C++标准的扩展。使用标准 C++编写的应用程序可以比较容易地从一种系统环境移植到另一种系统环境中,而且该语言是目前功能最强大的编程语言之一,因此该语言的使用范围非常广泛。然而,如果既想继续使用传统 C++提供的全部功能,又想拥有 C#般的编程效率,并且还能够使用 .NET 提供的强大的功能,就需要使用 C++/CLI 编写 .NET 应用程序。



1.1 .NET 与 C++/CLI

.NET Framework 在 Visual Studio 2010 开发平台及微软其他 .NET 开发产品中都是核心组件,主要由 CLR (Common Language Runtime, 公共语言运行时) 和 .NET Framework 类库两个核心部分组成。CLR 提供了用户程序执行的环境,而 .NET Framework 类库提供了在 CLR 中执行程序代码时所需的支持。在编写 .NET 程序时可以用 C++、C# 或者任何其他支持 .NET Framework 的语言编写,而这些语言都使用相同的 .NET 程序库。

1.1.1 .NET 框架概述

.NET 是 Microsoft 面向 XML Web 服务的平台。XML Web 服务允许应用程序,不管所采用的是哪种操作系统、设备或编程语言,均可以通过 Internet 进行通信和共享数据。

其中，.NET 框架（.NET Framework）是.NET 的核心。.NET 框架是由多个组件组成的庞大的框架，它执行应用程序和 Web 服务（包括类库，称为.NET 框架类库或 FCL），并提供安全性及许多其他的编程功能。使用.NET 开发的应用程序都需要在.NET 框架下运行。

.NET 框架的体系结构包括五大部分：程序设计语言及公共语言规范（CLS）、应用程序平台（ASP.NET 及 Windows 窗体等）、ADO.NET 及基础类库、公共语言运行时（CLR）和程序开发环境（Visual Studio.NET）。.NET 框架的体系结构如图 1.1 所示。

- 公共语言规范（Common Language Specification, CLS）：CLS 定义了各种语言间的互操作性规则，是许多应用程序所有互用语言的通用功能，如图 1.2 所示。
- 应用程序平台：该部分主要包括 ASP.NET 应用程序和 Windows Forms 应用程序，其中 ASP.NET 应用程序主要用于构建基于 Internet 的应用程序，而 Windows 窗体用于创建基于 CLR 的 Windows 图形用户界面的窗口应用程序。
- ADO.NET：ADO.NET 主要用于操作存储在关系型数据库管理系统（DBMS）中的数据，它提供了对关系数据、XML 和应用程序数据的访问，是.NET 框架中重要的组成部分。
- 基类库（Base Class Library）：.NET 框架的基类库提供许多类与接口，包括 ADO.NET、XML、IO、网络、调试、安全和多线程等，这些类库都是以命名空间的方式来组织的。
- 公共语言运行时（Common Language Runtime, CLR）：CLR 的作用是负责执行程序，并提供内存管理、线程管理、安全管理、异常处理、通用类型系统及生命周期监控等核心服务。
- 编程工具开发环境：Visual Studio.NET 是一套多语言系列的开发工具，使用 Visual Studio.NET 不仅可以创建 Windows 平台下的 Windows 应用程序和网络应用程序，还可以创建网络服务、智能设备应用程序和 Office 插件等。

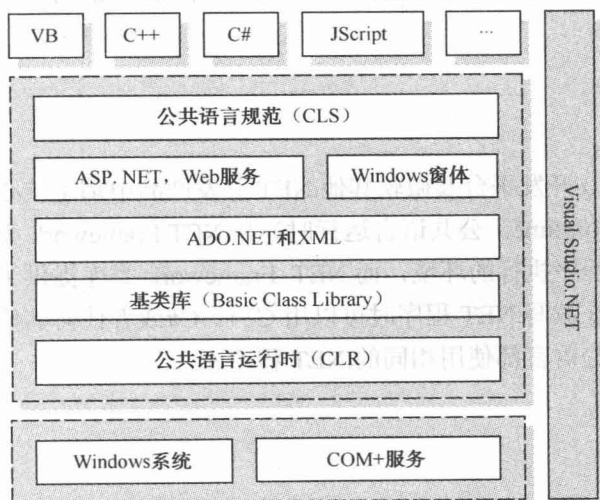


图 1.1 .NET 框架的体系结构

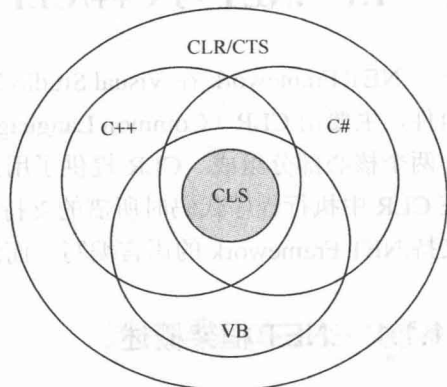


图 1.2 公共语言规范 CLS

随着.NET技术的不断发展,.NET Framework的发展也经历了几个阶段,从早期的.NET Framework 1.0、1.1发展到.NET Framework 2.0,标志着.NET技术走向成熟,功能也更加强大。在2008年,随着Microsoft推出Visual Studio 2008开发平台,.NET Framework又由2.0更新为3.0和3.5。.NET Framework 3.0、3.5分别在.NET Framework 2.0的基础上进行扩展,增加了很多新特性,如WCF、WPF、WF、LINQ和AJAX等。而目前.NET Framework最新版本为4.0,并被集成在Visual Studio 2010开发平台中。

1.1.2 公共语言运行时

公共语言运行时(Common Language Runtime, CLR)是为执行给定编译代码所需外部服务的集合。例如,若使用MFC建立一个应用程序时,那么该程序就需要MFC运行库;而对于Java应用程序,则需要依靠Java虚拟机等。

CLR是所有.NET应用程序运行时的环境,是所有.NET应用程序都使用的编程基础。CLR也可以看做是一个在执行时管理代码的代理,管理代码是CLR的基本原则,能够被管理的代码成为托管代码,反之称为非托管代码。

CLR中最重要的部分是由mscoree.dll库(又称为公共对象运行库执行引擎)表示的。当应用程序引用一个程序集时,首先自动加载mscoree.dll库,然后由它负责将需要的程序集导入内存。

在.NET框架之上,无论采用哪种编程语言编写的应用程序,都首先被编译成中间语言IL(Microsoft Intermediate Language, 微软中间语言,简称IL),然后IL再次经过一个JIT(Just In Time, 即时编译)编译器编译成为本地CPU可执行的机器码。.NET编译器和.NET执行引擎的工作流程图如图1.3所示。

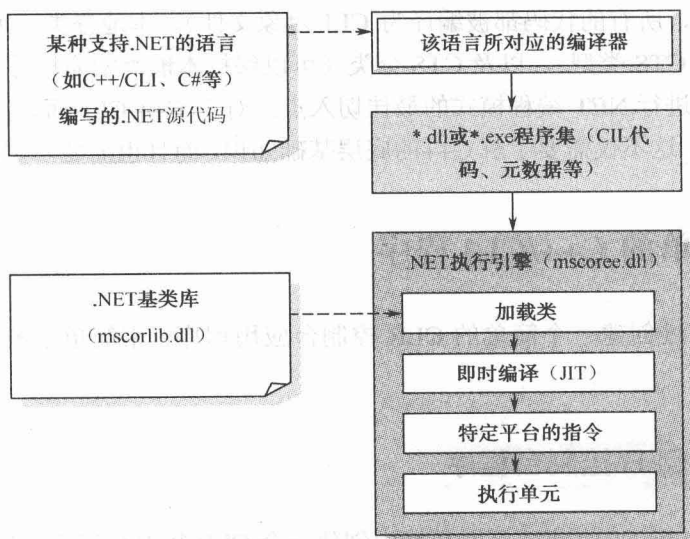


图 1.3 .NET 编译器和.NET 执行引擎的工作流程图



在 Windows 平台中, CLR 带有三个不同的 JIT 编译器: 主编译器、PREJIT 和 ECONOJIT。其中, 主编译器是默认的编译器, 它主要进行数据流分析并输出经过优化的本地代码, 所有的中间代码指令均可被它处理; PREJIT 编译器是建立在主 JIT 编译器之上, 并且每当一个 .NET 组件被安装时它就运行; ECONOJIT 编译器能够在并不充分优化的前提下, 快速完成 IL 代码到本地码的转换, 编译速度与运行速度都非常快。

1.1.3 C++/CLI 概述

Visual C++ 开发小组在对 .NET 和 C++ 经过仔细考量之后, 决定在 Visual C++ 2005 中重新设计对通用语言运行时库 (CLR) 的支持。此项重新设计被称为 “C++/CLI”, 它为使用及编写 CLR 类型提供更自然的语法。

公共语言运行时库 (CLR) 包括了一组规范, CLR 是 Microsoft .NET 的基础, 也是 CLI 的 Microsoft 版本。C++/CLI 语言设计的目标是为了对 CLI 提供更自然的 C++ 支持, 而 Visual C++ 2005 以及后续版本的编译器则在 CLR 上都实现了 C++/CLI。

C++/CLI 是一种用来代替 C++ 托管扩展的新的语言规范, 它重新简化了 C++ 托管扩展的语法, 并提供了更好的代码可读性。和微软 .NET 的其他语言一样, 微软向 ECMA 提交了 C++/CLI 的标准。在 2005 年 11 月, ECMA 正式推出了 C++/CLI 的标准 ECMA-372。现在可以在 Visual C++ 2005、Visual C++ 2008 及 Visual C++ 2010 集成开发环境上使用 C++/CLI 进行开发。

C++/CLI 代表托管和本地编程的结合。在反复过程中, 这种结合已经通过源代码级相对独立但又相互平等的组件和二进制元素得到了完成, 包括混合模式 (本地和 CTS 类型的源代码级混合, 还有一个本地及 CLI 对象文件的二进制混合), 纯模式 (本地和 CTS 类型的源代码级混合, 所有的代码都被编译为 CLI 对象文件), 本地分类 (可以通过一个特定的打包类来保持 CTS 类型), 以及 CTS 分类 (可以保持本地类型为指针)。

C++/CLI 是进行 .NET 编程模式的最佳切入点。对于 C++/CLI 而言, 需要有一个来自 C++ 的迁移路径, 这不仅需要了解 C++ 的底层基础知识, 而且也需要一定的 C++ 编程经验。



1.2 简单的 C++/CLI 程序

在本节中通过创建一个简单的 CLR 控制台应用程序, 来简单了解 C++/CLI 的语法结构。

1.2.1 创建 CLR 控制台程序

在 Visual Studio 2010 集成开发环境中创建一个 CLR 控制台应用程序。该程序将从控制台读取用户输入的值, 并根据该值来计算圆的面积, 并输出到控制台上。

【例 1.1】 创建 CLR 控制台程序。

(1) 启动 Visual Studio 2010 集成开发环境，在菜单栏中选择“文件”→“新建”→“项目”菜单命令，弹出如图 1.4 所示的“新建项目”对话框。

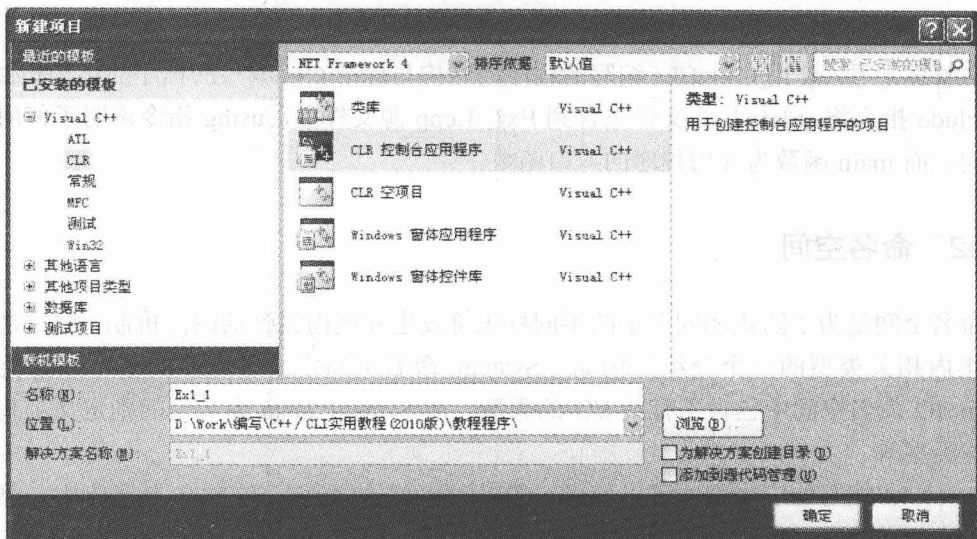


图 1.4 “新建项目”对话框

(2) 在对话框的左侧选择“Visual C++”节点下的“CLR”节点，并在对话框的中部选择“CLR 控制台应用程序”项。在“名称”栏中输入 Ex1_1，并在“位置”栏中指定该项目保存的位置，可以单击“浏览”按钮选择其他保存路径。

(3) 单击“确定”按钮后，集成开发环境会自动创建 Ex1_1 项目，并自动打开源代码编辑窗口。此时可以在 Ex1_1.cpp 源文件的 main 函数中添加计算圆面积的代码。代码如下：

```
// Ex1_1.cpp : 主项目文件
#include "stdafx.h"
using namespace System;

int main(array<System::String ^> ^args)
{
    Console::Clear();
    Console::Write(L"输入圆的半径: ");

    double r = Double::Parse(Console::ReadLine());
    double area = 3.14159 * r * r;

    Console::WriteLine(L"半径为{0}的圆面积为: {1}", r, area);
    return 0;
}
```

(4) 选择“生成”→“生成解决方案”菜单命令以编译项目，并选择“调试”→“开始执行（不调试）”菜单命令以运行项目。Ex1_1 项目运行的结果如图 1.5 所示。

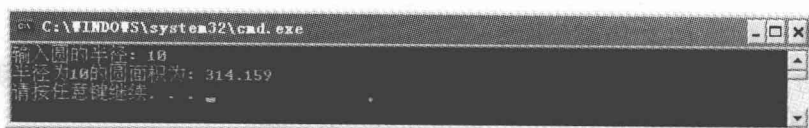


图 1.5 Ex1_1 项目的运行结果

从整体上看,使用 C++/CLI 编写的程序的结构与标准 C++编写的程序的结构相似,其中 `#include` 指令将 `stdafx.h` 头文件包含到 `Ex1_1.cpp` 源文件中, `using` 指令声明了使用的命名空间,而 `main` 函数为应用程序的入口函数。

1.2.2 命名空间

命名空间是为了防止相同名字的不同标识符发生冲突而设计的隔离机制,同时是一个程序集内相关类型的一个分组。例如, `System` 命名空间定义了许多基本数据类型, `System::IO` 命名空间中包含了有关 I/O 的类型,而 `System::Data` 命名空间则定义了基本的数据库类型等。

C++/CLI 编写的应用程序可以使用 .NET 基类库中的类型。在 .NET 基类库中,最基本的命名空间是 `System`,其中提供了大量核心的类型,并在程序中会经常地使用。 .NET 常用命名空间及其说明如表 1.1 所示。

表 1.1 .NET 常用命名空间及其说明

.NET 命名空间	说 明
<code>System</code>	包含用于定义常用值和引用数据类型、事件和事件处理程序、接口、特性和处理异常的基础类和基类
<code>System::Collections</code> <code>System::Collections::Generic</code>	这些命名空间定义了一些集合容器对象,还有一些基类型和接口,可以允许用户创建强类型的集合
<code>System::Data</code> <code>System::Data::Odbc</code> <code>System::Data::OracleClient</code> <code>System::OleDb</code> <code>System::SqlClient</code>	这些命名空间包含组成大部分 ADO.NET 结构的类,其中封装用于 ODBC/Oracle/OleDb 和 SqlServer 的 .NET 数据提供程序的类,该提供程序描述用来访问托管空间中的 ODBC/Oracle/OleDb 和 SqlServer 数据源的类集合
<code>System::Diagnostics</code>	定义了许多类型,可以通过程序调试并跟踪代码的执行
<code>System::Drawing</code> <code>System::Drawing::Drawing2D</code> <code>System::Drawing::Printing</code>	提供对 GDI+ 基本图形功能的访问,其中提供高级的二维和矢量图形功能,以及提供与打印相关的服务
<code>System::IO</code>	包含允许对数据流和文件进行同步和异步读写的类型
<code>System::Net</code>	为当前网络采用的多种协议提供简单的编程接口
<code>System::Threading</code>	提供支持多线程编程的类和接口
<code>System::Windows::Forms</code>	包含用于创建基于 Windows 的应用程序的类,这些应用程序可以充分利用 Windows 操作系统中的丰富的用户界面功能