附录A　C++的.NET可管理扩展

Microsoft的.NET框架定义了支持开发和运行高度分布式、基于组件的应用程序的环境。它使我们能够混合使用不同的计算机语言，它也提供了安全性、程序可移植性和Windows平台使用的通用编程模型。尽管.NET框架是在最近才增加到计算中的，它是一种环境，在可预见的将来，许多C++程序员会在其中进行他们的工作。

Microsoft的.NET框架提供了可管理的环境，可以监视程序的执行。以.NET框架为目标的程序不能编译成可执行目标码，但是可编译成Microsoft Intermediate Language(MSIL)，它是在通用语言运行环境（Common Language Runtime，CLR）的控制下执行的。可管理执行是一种支持.NET框架提供的主要优点的机制。

要利用.NET可管理的执行机制，C++程序必须使用一组由Microsoft定义的非标准扩展关键字和预处理程序指令。重要的是要理解这些扩展不是ANSI/ISO标准C++定义的。因此，包含它们的代码不可移植到其他的环境中。

描述利用.NET框架所需要的.NET框架或C++编程技巧超出了本书讲述的范围（全面讲述.NET框架和如何创建适用于它的C++代码就需要整整一大本书）。然而，简短概述一下C++的.NET可管理扩展对在.NET环境下工作的程序员是有帮助的。当然，这里假定读者对.NET框架有基本的理解。

A.1　.NET关键字扩展

为了支持.NET可管理执行环境，Microsoft给C++语言增加了下面的关键字：

__abstract	__box	__delegate
__event	__finally	__gc
__identifier	__interface	__nogc
__pin	__property	__sealed
__try_cast	__typeof	__value

下面简单描述一下这些关键字。

A.1.1　__abstract

__abstract和__gc一起使用，用以规定一个抽象可管理类。不可以创建__abstract类的对象。被指定为__abstract的类不需要包含纯虚函数。

A.1.2　__box

__box把一个值封装在一个对象内。封装（Boxing）使得值类型可以被某种代码使用，这种代码要求一个从System::Object派生出的对象，System::Object是所有.NET对象的基类。

A.1.3 __delegate

__delegate指定一个委托，该委托封装一个指向可管理类（即由__gc修饰的类）内一个函数的指针。

A.1.4 __event

__event指定一个表示事件的函数。只规定了这个函数的原型。

A.1.5 __finally

__finally是后加到标准C++异常处理机制中的。我们使用它来指定当去掉try/catch时将要执行的代码块。它不管什么条件引起try/catch块终止。无论何种情况，都会执行__finally块。

A.1.6 __gc

__gc指定一个可管理类。这里，“gc”表示“garbage collection”（即无用单元收集）它也表示当不再需要这个类的对象时，它们被自动进行无用单元收集。当对象的引用不存在时，就不再需要这个对象了。__gc类的对象必须使用new来创建。数组、指针和接口也可被指定为__gc。

A.1.7 __identifier

__identifier允许把一个C++关键字用做一个标识符。这是一个特殊用途的扩展，不被大多数程序所使用。

A.1.8 __interface

__interface指定一个用做接口的类。在接口中，不包括函数体。接口中的所有函数都是隐含的纯虚函数。因此，一个接口本质上是一个抽象类，其中的函数都没有实现。

A.1.9 __nogc

__nogc指定一个不可管理的类。因为这是默认时创建的类类型，所以通常不使用__nogc关键字。

A.1.10 __pin

使用__pin来指定一个指针，该指针固定它所指向的对象在内存中的位置。因此，无用单元收集器不能在内存中移动所固定的对象。结果，无用单元收集不能使一个由__pin修饰的指针无效。

A.1.11 __property

__property指定一个性质，该性质是一个得到或设置成员变量值的成员函数。性质提供了一个控制对私有或受保护数据的访问的方便方法。

A.1.12 __sealed

__sealed防止它修饰的类被继承。也可以使用它来指定一个虚函数不能被覆盖。

A.1.13 __try_cast

__try_cast尝试把一种表达式强制转换成另一种表达式。如果转换失败，抛出一个System::InvalidCastException类型的异常。

A.1.14 __typeof

__typeof获得一个封装所给类型的类型信息的对象。这个对象是System::Type的一个实例。

A.1.15 __value

__value规定了一个被表示为值类型的类，值类型包含它自己的值。这不同于__gc，__gc必须使用new分配存储空间。值类型不受无用单元收集的影响。

A.2 预处理程序扩展

为支持.NET，Microsoft定义了#using预处理程序指令，用于把元数据导入到程序中。元数据包含独立于具体计算机语言的类型和成员信息。因此，元数据帮助支持多种语言混合编程。所有可管理的C++程序必须导入<mscorlib.dll>，它包含.NET框架的元数据。Microsoft定义了两个与.NET框架有关的范型（范型和#pragma预处理指令一起使用）。第一个是managed，它规定了可管理代码。第二个是unmanaged，它规定了不可管理（即本地的）代码。这些范型可以在程序内使用，以便有选择地创建可管理和不可管理的代码。

A.3 attribute属性

Microsoft定义了attribute，这是用于声明另一个属性的属性。

A.4 编译可管理的C++

在写本书时，以.NET框架为目标的、惟一可用的编译器是由Microsoft的Visual Studio .NET提供的。要编译可管理的代码程序，必须使用/clr选项，/clr选项的目标是通用语言运行环境代码。

附录 B　C++ 和机器人时代

笔者很长时间以来就对机器人、特别是机器控制语言感兴趣。事实上，几年前，笔者就设计和实现了在小的教育用机器人上使用的工业机器控制语言。尽管已不再专门工作在机器人领域，笔者仍然对此抱有很浓厚的兴趣。这些年来，机器人的功能（和控制它们的代码）有了很大的飞跃。我们现在正处在机器人时代的开始，已经有了可以除草和清洁地板的机器人，机器人也可以用来组装汽车及工作在对人类有危险的环境中，战争用机器人也正在变成现实。许多机器人应用程序正在开发过程中。随着机器人变得越来越普遍、越来越多地出现在我们的日常生活中，日益增多的程序员会发现自己正在编写机器人控制代码，并且其中的大部分代码是用 C++ 编写的。

C++ 是进行机器人编程时一种很自然的选择，因为机器人要求高效、高性能的代码。对于低级发动机控制例程，以及像视觉处理（在此速度是相当重要的）这样的事情，尤其如此。尽管机器人的子系统的某些部件，例如自然语言处理器，可以用像 C# 这样的语言来编写，低级代码几乎总是要用 C++ 来编写。C++ 和机器人总是并肩出现。

如果你对机器人感兴趣，特别是如果你对创建自己的试验用机器人感兴趣，那么，你会发现图 B.1 所示的机器人很有趣，这是笔者的试验用机器人。这个机器人有几件事特别有趣。首先，它包含一个微处理器，用于提供发动机控制和传感器反馈。第二，它包含一个 RS-232 接收器，用于接收来自主计算机的指令并返回结果。这个方法使远程计算机能够提供强大的处理能力（这种能力是机器人所需要的），而没有给机器人本身增加重量。第三，它包含一个连接到无线视频发送机的摄像机。

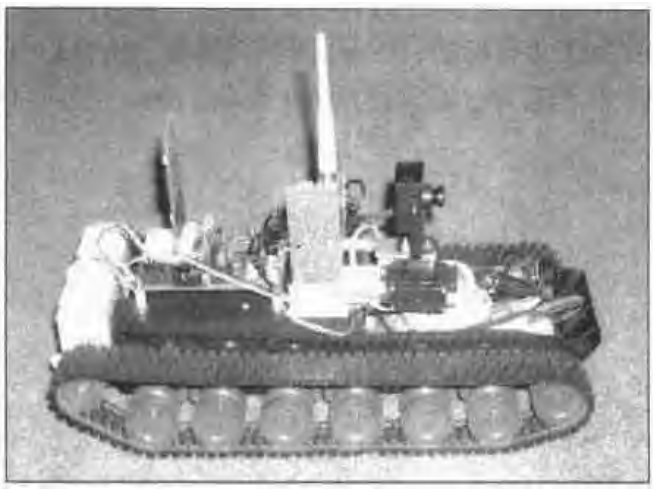

图 B.1　一个简单、有效的试验型机器人（Ken Kaiser 摄）

这个机器人是构建在 Hobbico M1 Abrams R/C 坦克底盘上的（我发现 R/C 坦克和汽车的底盘经常用做机器人的基座）。我把大多数内部部件从坦克中移走，包括接收器和速度控制器，但是保留了发动机。Hobbico 坦克很适宜于机器人平台，因为它相当大，发动机很好，载重大且坦克针（tread）不会掉落。还有，通过使用坦克针，机器人的转动半径为 0 且能够在不平坦的

地面上行走。底盘大约为18英寸长，8英寸宽。

一旦底盘空了，就加入下面的组件。要提供现场控制，使用一个BASIC Stamp 2，它是一个简单、然而强大的微处理器，由Parallax. Inc. (www.parallaxinc.com)生产。RS-232接收器也是Parallax公司生产的，就像摄像机和转换器一样。无线RS-232接收器和视频转换器大约为300英尺的范围。我也为坦克发动机增加了电子速度控制器，它们是高性能的R/C汽车所用的类型的，由BASIC Stamp微处理器控制。

下面是机器人工作的原理。远程计算机运行主要的机器人控制程序，这个程序处理所有"重要"的处理任务，如视觉、导航和空间定位。它也学习一系列的动作，然后重放它们。远程计算机把控制动作的指令传输到机器人处（通过无线RS-232）。BASIC Stamp接收那些指令并开始行动。例如，如果接收了"向前移动"的命令，BASIC Stamp向连接到发动机的电子速度控制器发送合适的信号。当机器人完成一个命令时，它返回一个确认码。因此，在远程计算机和机器人之间的通信是双向的，可以确认成功完成的每一个命令。

因为对机器人的主要处理出现在远程计算机上，所以对我能够做的处理数量并没有严格的限制。例如，在写本书时，这个机器人通过使用它的视觉系统，能够跟踪一个对象。这种能力要求适中的处理量，很难实现。

最近，我开始对将要增加到机器人上的机器人手臂进行研究。机器人手臂的原型如图B.2所示。尽管试验人员可以得到几种商用的机器人手臂，我决定自己创建，因为我想要一个更强壮、能举起更重的对象的机器人手臂。这个手臂使用装在它的基座上的stepper发动机转动一个长的像针一样的镙丝（它打开和关闭抓具）。这个方法允许准确地移动，且强度相当大。手臂由它自己的Stamp控制。因此，主机器人控制器简单地把手臂命令传给第二个Stamp。这允许机器人和手臂并行操作，防止了主机器人控制器陷于损坏状态。

图B.2　一个机器人手臂的原型

尽管主要的机器人控制代码将永远保留在C++中，我正在试验把两个子系统、包括RS-232通信例程迁移到C#中。C#提供了一个到IP数据转换的方便的接口，并且能够通过Internet从远程地点控制机器人是一个让人振奋的想法。

后 记

C++语言是在C语言的基础上、吸收了BCPL和Ada等语言中的精华而逐渐发展起来的一种通用的程序设计语言。它的出现是为适应开发和维护中等和大规模复杂应用软件的需要，其目标是为程序员提供一个好的编程环境，以生产模块化程度高、可重用、可维护的程序，同时它强调了代码的有效性和紧凑性。

自1983年贝尔实验室公布C++以来，C++就在计算机界引起了广泛的重视。在20世纪80年代末ISO和ANSI开始对C++进行标准化的工作，产生了所谓的ISO/ANSI C++。C++的主要目标是支持面向对象的程序设计方法。市面上介绍C++的书很多，这些书的内容可以说是"大而全"。C++语言的入门并不难，但是正如许多人认为的那样，掌握它的最难之处是用其灵活高效地开发出实用的软件来。

本书是由编程语言大师Herbent Shildt撰写的。书中全面描述和演示了定义C++语言的关键字、语法、函数、类和特征。更准确地讲，本书全面描述了标准C++语言。标准C++是由ANSI/ISO C++标准定义的C++版本，也是为所有主要编译器，包括Microsoft的Visual C++和Borland的C++ Builder所支持的C++版本。因此，本书中的内容适用于所有现代编程环境。

本书主要由周志荣、朱德芳、于秀山、王丽丽、吴习东、刘未名、贾晓芳、卞昭华、肖广慧、张宇翻译，参加翻译工作的还有王玉琴、宋平、范永斌、张强、袁源、赵小龙、张文、胡明、苏小林、王金全、李云静、舒禾、叶柏青、唐化、周蓓、刘燕京等。

译者力求反映原书的特点和风貌，但由于水平有限，不当和疏漏之处在所难免，敬请读者批评指正。

后 记

C++语言是在C语言的基础上、吸收了BCPL和Ada等语言中的精华而逐渐发展起来的一种通用的程序设计语言。它的出现是为适应开发和维护中等和大规模复杂应用软件的需要，其目标是为程序员提供一个好的编程环境，以生产模块化程度高、可重用、可维护的程序，同时它强调了代码的有效性和紧凑性。

自1983年贝尔实验室公布C++以来，C++就在计算机界引起了广泛的重视。在20世纪80年代末ISO和ANSI开始对C++进行标准化的工作，产生了所谓的ISO/ANSI C++。C++的主要目标是支持面向对象的程序设计方法。市面上介绍C++的书很多，这些书的内容可以说是"大而全"。C++语言的入门并不难，但是正如许多人认为的那样，掌握它的最难之处是用其灵活高效地开发出实用的软件来。

本书是由编程语言大师Herbent Shildt撰写的。书中全面描述和演示了定义C++语言的关键字、语法、函数、类和特征。更准确地讲，本书全面描述了标准C++语言。标准C++是由ANSI/ISO C++标准定义的C++版本，也是为所有主要编译器，包括Microsoft的Visual C++和Borland的C++ Builder所支持的C++版本。因此，本书中的内容适用于所有现代编程环境。

本书主要由周志荣、朱德芳、于秀山、王丽丽、吴习东、刘未名、贾晓芳、卞昭华、肖广慧、张宇翻译，参加翻译工作的还有王玉琴、宋平、范永斌、张强、袁源、赵小龙、张文、胡明、苏小林、王金全、李云静、舒禾、叶柏青、唐化、周蓓、刘燕京等。

译者力求反映原书的特点和风貌，但由于水平有限，不当和疏漏之处在所难免，敬请读者批评指正。

后 记

C++语言是在C语言的基础上、吸收了BCPL和Ada等语言中的精华而逐渐发展起来的一种通用的程序设计语言。它的出现是为适应开发和维护中等和大规模复杂应用软件的需要，其目标是为程序员提供一个好的编程环境，以生产模块化程度高、可重用、可维护的程序，同时它强调了代码的有效性和紧凑性。

自1983年贝尔实验室公布C++以来，C++就在计算机界引起了广泛的重视。在20世纪80年代末ISO和ANSI开始对C++进行标准化的工作，产生了所谓的ISO/ANSI C++。C++的主要目标是支持面向对象的程序设计方法。市面上介绍C++的书很多，这些书的内容可以说是"大而全"。C++语言的入门并不难，但是正如许多人认为的那样，掌握它的最难之处是用其灵活高效地开发出实用的软件来。

本书是由编程语言大师Herbent Shildt撰写的。书中全面描述和演示了定义C++语言的关键字、语法、函数、类和特征。更准确地讲，本书全面描述了标准C++语言。标准C++是由ANSI/ISO C++标准定义的C++版本，也是为所有主要编译器，包括Microsoft的Visual C++和Borland的C++ Builder所支持的C++版本。因此，本书中的内容适用于所有现代编程环境。

本书主要由周志荣、朱德芳、于秀山、王丽丽、吴习东、刘未名、贾晓芳、卞昭华、肖广慧、张宇翻译，参加翻译工作的还有王玉琴、宋平、范永斌、张强、袁源、赵小龙、张文、胡明、苏小林、王金全、李云静、舒禾、叶柏青、唐化、周蓓、刘燕京等。

译者力求反映原书的特点和风貌，但由于水平有限，不当和疏漏之处在所难免，敬请读者批评指正。