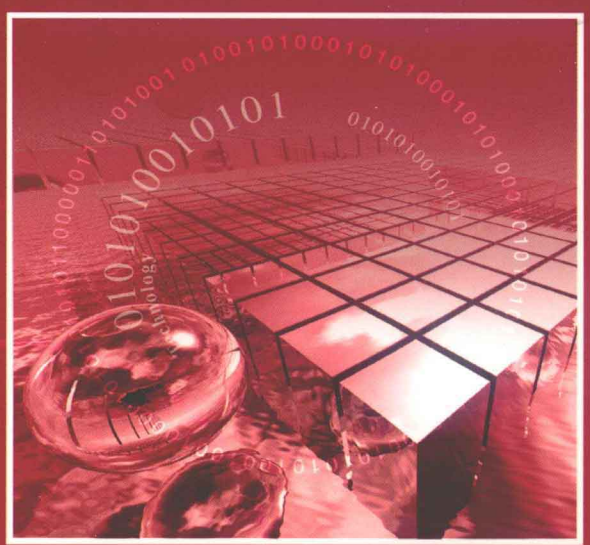
普通高等教育“十一五”国家级规划教材

普通高等教育“十一五”计算机类规划教材

C#编程和.NET框架

崔建江 主编



机械工业出版社
CHINA MACHINE PRESS

普通高等教育“十一五”国家级规划教材

C#编程和.NET框架

主　编　崔建江
副主编　贾　同　张云洲　邓天民
主　审　范立南　胡琨元

机械工业出版社

本书是普通高等教育“十一五”国家级规划教材之一，介绍C#语言编程和其开发平台. NET框架的主要内容。全书共分13章。第1章介绍本课程的预备知识、C#和. NET的概要内容。第2章至第5章介绍C#相关的知识，包括C#的开发环境以及编译调试方法、C#语言基础、面向对象的C#程序设计和C#进阶。第6章介绍. NET框架。第7章讲解C#窗体程序设计。第8章介绍C#流与文件操作。第9章介绍ADO. NET程序开发。第10章介绍ASP. NET程序开发。第11章介绍Web服务和访问Internet。第12章介绍C#高级应用，如C#创建和调用DLL、C#串行通信实现等。第13章给出了一个完整的基于. NET开发的实例。

本书是在作者多年讲授C#课程的讲义基础上整理而成的，包含多年的实际经验。本书力求内容组织合理、难易程度适当，并考虑了学校讲授时的课时限制情况，因此比较适合作为普通高等学校各类专业本科生和研究生学习C#和. NET的教材或参考书。本书定位为相关知识的入门读物，因此也可供一般读者学习和掌握C#语言编程和. NET框架的入门参考。学习本书的读者最好具备一定的C/C ++的知识，以期达到较好的学习效果。

图书在版编目（CIP）数据

C#编程和. NET框架/崔建江主编. —北京：机械工业出版社，2011. 12
普通高等教育“十一五”国家级规划教材
ISBN 978-7-111-37341-4

Ⅰ. ①C… Ⅱ. ①崔… 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（2012）第015894号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）
策划编辑：闫晓宇 责任编辑：闫晓宇
版式设计：霍永明 责任校对：陈秀丽 李锦莉
封面设计：张 静 责任印制：杨 曦
北京京丰印刷厂印刷
2012年4月第1版 · 第1次印刷
184mm×260mm · 24.25印张 · 602千字
标准书号：ISBN 978-7-111-37341-4
定价：48.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
社服务中心：（010）88361066
销售一部：（010）68326294
销售二部：（010）88379649
读者购书热线：（010）88379203

网络服务
门户网：http://www.cmpbook.com
教材网：http://www.cmpedu.com

前　言

我国软件产业持续保持高速发展，已成为我国电子信息产业中增长最快、潜力最大的领域之一。软件产业的产值由2001的750亿元跃升到2010年的1.3万亿元，增长了十几倍。

但是，在这样的背景之下，国内调查机构最近的调查结果显示，我国IT人才的缺口很大，IT人才紧缺已成为制约IT行业发展，尤其是软件发展的最大瓶颈。同时，巨大的人才空缺并没有缓解大量学生就业难的问题。计算机基础教育实质上是应用教育，应用教育必须面向市场需求。目前学校教育与市场需求存在着矛盾，企业与学校之间缺乏沟通，学校培养出来的学生不是企业想要的：学校未能扩展课程，涵盖现在技术发展所需知识基础，而企业不愿意雇佣并不具备岗位基本技能的员工，增加训练成本。正是这样的矛盾，使得一方面是企业求贤若渴，另一方面又是毕业生就业无门。2002年，教育部提出要从院校开始培养企业需要的人才，通过企业与院校的合作，将最新的IT技术融入教学，为企业培养优秀的人才。

IT技术革命从第一代的基于文本、第二代的基于图形，已经进入了基于Web服务的第三代，而目前学校所设课程大多仍然停留在第一和第二代水平，从教育的源头就不符合社会对人才的需求。为此，微软（2005年）基于全球3万份岗位技能分析的结果，得出了27个IT岗位，并根据每个岗位的技能分析设计出了一套面向涵盖目前IT岗位技能的完整课程体系。与该IT课程体系相比较，作为IT人才培养的基础部分的有关.NET框架和C#编程语言课程，在许多院校目前都没有开设。

.NET框架是微软在2000年专业开发者会议上提出的发展中的开发平台，这是一个革命性的应用程序开发平台，该平台建立在开放的Internet协议和标准之上，采用了许多新的工具和服务，用于计算和服务，创建XML Web服务（下一代软件）平台，该平台将信息、设备和人以一种统一的、个性化的方式联系起来。

在该平台中，C#作为微软面向对象的下一代应用平台的核心语言，能够让开发人员在.NET平台上快速开发应用程序，可以说，C#是面向.NET框架的、可以兼顾开发能力和效率的、与当前的Web应用结合较好的、全新的开发工具和首选开发语言。C#为程序员提供了开发Web应用程序所需要的强大而灵活的功能，未来大量“.NET”平台的应用将由C#开发，C#将是未来开发企业级分布式应用程序的首选。

鉴于.NET框架以及核心编程语言C#在微软的大力推广下将会成为未来影响人类生活的最先进技术平台之一，并且是今后IT人才培养的基础，因此，学习和开设相关课程就显得十分必要和迫切。本书就是在此背景下应运而生的。

微软正在五个方面构建.NET平台，即工具、服务器、XML Web服务、客户端和.NET体验，其新的理念涉及计算机技术的方方面面，很多内容都可以作为一个专题课程开设。为此，作者经过大量的相关技术分析和国内外书籍分析，精选了具有代表性的核心内容，包括：C#语言及其开发环境、.NET框架、ADO.NET和ASP.NET等。另外，本教材还精选了Web服务、访问Internet等内容，为学生奠定了学习相关知识的入门基础。

.NET包含了丰富的计算机知识，作为教材，本书本着针对基础、展现核心、有效实用的原则组织内容。由于本书的先修内容应包括C/C++和程序设计的基础知识，因此相同的知识内容本书大多不再重复叙述，使用本书教学时可以根据学生情况进行补充。

本书在使用时可分几个层次进行：第1章至第6章为基本内容，可用32学时完成，另外安排8学时的实验；当教学课时为40学时时，可增加第8章和第9章内容，将本书的核心内容教授完，并安排至少12学时的实验；第7章、第10章、第11章和第12章可根据需要选讲；第13章可作为实验作业或自学内容，通过实例掌握基于C#和.NET的软件开发方法。

本书是作者在2004年教育部和微软（中国）长城计划中精品课程计划支持下编写并讲授多轮的讲义的基础上撰写的。参加讲义代码验证、资料整理和教辅材料工作的有赵伟、楚峰、庄严和龙丽。

本书撰写工作的具体分工为：第1章至第5章由崔建江撰写，第6章由崔建江和贾同撰写，第9章、第10章、第13章由贾同撰写，第7章、第8章、第12章由张云洲撰写，第11章由邓天民撰写。崔建江负责全书的框架设计和统稿工作。本书由范立南、胡琨元主审。参与资料整理、勘误、代码验证工作的有赵勇、孟祥军、张亮、丁万强、于晓升、司鹏举、陈世峰、迟剑宁、李翠娟、李思远。

本课程的建议工作得到了有关领导和教师的大力支持与帮助，在此一并致谢。

作者水平所限，书中难免存在漏误之处，还望广大读者批评指正。

崔建江
于东北大学

目　录

第1章 绪 论

随着计算机软件技术的发展，计算机软件的应用范围越来越广，复杂程度也越来越高。软件的功能也发生了巨大的变化，从开始的只能进行一些复杂的数学运算，到能够完成越来越复杂的各种其他功能，并且呈现了网络化、智能化、协同化等特点。随之而来的，实现软件设计的计算机语言也正逐步地发生相应的变化。本章首先介绍计算机语言的基础知识，使读者对计算机语言的发展有个初步了解。然后对 C#语言和 .NET 框架做基本介绍，为后续的学习打下基础。

1.1 计算机语言概述

计算机语言指用于人与计算机之间通信的语言，程序员利用计算机语言根据算法编写指令，使计算机完成相应的功能。计算机系统的一大特征是指令通过一种语言传达给机器。为了使电子计算机进行各种工作，就需要有一套用以编写计算机程序的数字、字符和语法规则，由这些数字字符和语法规则组成计算机的各种指令（或各种语句）即计算机能接受的语言。计算机每做的一次动作、一个步骤，都是按照已经用计算机语言编好的程序来执行。程序是计算机要执行的指令的集合，而程序全部都是用人们所掌握的语言来编写的。

从 1946 年世界上诞生第一台计算机起，在六十余年间，计算机技术迅速发展。随着计算机技术的不断发展，计算机语言的发展也相当迅速，目前已有数百种计算机语言供程序员编程使用。程序设计语言的发展经历了四个阶段：①机器语言；②汇编语言；③面向过程的高级语言；④面向对象的程序设计语言。其中，机器语言和汇编语言为面向机器的低级语言，而面向过程的高级语言和面向对象的程序设计语言都是高级语言。

1.1.1 机器语言

20 世纪 50 年代开始出现面向机器的机器语言和汇编语言。由计算机硬件系统可以识别的二进制指令组成的语言称为机器语言。在计算机发展的初期，软件工程师们只能用机器语言来编写程序。

计算机是由各种存储器件组成的，其只能直接理解自己的“机器语言”。作为每种计算机的“天生语言”，机器语言在计算机硬件设计时就已经定义。机器语言通常由一组数字串组成，最终简化为 0 和 1 两种状态的数字串，这些数字指示计算机如何进行操作。机器语言依赖于机器，每种特定的机器语言只能使用在该种特定类型的计算机上。

下面以一个具体的实例来说明，其功能是完成一个非常简单的加法，即 A + B = C：

+1300042774

+1400593419

+1200274027

从中可以看出，机器语言对于人来说是难以理解的，这样的语言对于程序员编写和理解

程序是十分困难的。另外由于通篇是数字，可读性太差，编程不方便，指令难记，容易出错且不易修改。但是，其优点也是很显著的：能被计算机直接识别和执行，执行速度快。这一阶段，在人类的自然语言和计算机编程语言之间存在着巨大的鸿沟。

1.1.2 汇编语言

随着计算机的迅速发展及普及，人们愈发感到机器语言的枯燥和效率低下，并且十分容易出错。这时，程序员开始使用类似英语的缩写来代表计算机的基本操作，而不使用计算机可以直接理解的数字串。这些缩写构成了“汇编语言”的基础，同时使用“翻译程序”将汇编语言转化为机器语言供计算机识别。汇编语言就是符号化的机器语言，即将机器指令映射为一些可以被人读懂的助记符，如用记忆符 ADD 代表加法指令、OUT 代表输出指令等。

同样以上述加法为例，可以看出汇编语言与机器语言之间的差别：

```
MOV AX, A;
ADD AX, B;
MOV C, AX;
```

显然，这样的代码更利于人的理解，编程相对较为方便。但是，计算机不能识别汇编语言，因此要使用编译器将其转化为机器语言，也即需要“翻译”。汇编语言仍脱离不开具体机器的指令系统，它所用的指令符号与机器指令基本上是一一对应的，编程效率不高，编程人员需要熟悉计算机结构，因此一般人很难使用。此时编程语言与人类自然语言间的鸿沟略有缩小，但仍与人类的思维方式相差甚远，因为它的抽象层次太低，程序员需要考虑大量的机器或硬件细节。

1.1.3 面向过程的高级语言

尽管随着汇编语言的出现，计算机的使用迅速增加，软件编程工作也得到大幅改善，但是为了完成一件简单的任务（如循环、分支等），汇编语言仍然需要很多条指令，并且由于程序指令过于繁琐，仍容易造成错误和混淆。为了加快编程的过程，20 世纪 60 年代高级语言的出现大大简化了程序设计，可以用简单的语句来完成大量的任务。高级语言的指令和日常英语非常相似，并且还包括了常见的数学符号。

下面的指令是用高级语言编写的实现上述加法功能的指令：

```
C = A + B;
```

显然，与机器语言和汇编语言相比，高级语言更易于理解和编程，更受程序员欢迎。但与汇编语言一样，这样的程序代码计算机是不能识别的，需要使用编译器将其转化为机器语言。

高级语言与人类自然语言和数学式子相当接近，而且不依赖于某台机器，通用性好，如 C、FORTRAN、BASIC、PASCAL 等，都是面向过程的语言。

与以往的计算机语言比较，面向过程的语言编程十分方便，不必关心机器的细节，提高了语言的抽象层次，程序中可以采用具有一定含义的数据命名和容易理解的执行语句，这使得在书写程序时可以联系到程序所描述的具体事物。

但是，面向过程的语言存在仍然需要把高级语言编写的程序（称源程序）翻译成机器语言程序（称目标程序）。另外，开发的程序可重用性差、数据安全性差，并且难以开发图

形界面。

1.1.4　面向对象的高级语言

20 世纪 80 年代，人们提出了面向对象的程序设计方法（Object Oriented Programming，OOP）：采用面向对象思想的程序设计高级语言。

面向对象的思想：将客观事物看做具有属性和方法的对象，对象与对象之间通过消息（称做事件）进行通信，消息激发对象做出相应的反映。客观世界或其中一部分可看做是由各种对象组成的一个运动的有机整体。

面向对象的高级语言，和以往语言相比具有显著优点：面向对象的程序设计更为符合人的自然思维方式；设计开发的程序模块间的关系更为简单，程序模块的独立性、数据的安全性就有了良好的保障；通过继承、多态与封装，可以大大提高程序的可重用性，使得软件的开发和维护都更为方便。

1.1.5　高级编程语言的发展

高级编程语言的种类繁多，主要代表有：①面向过程的编程语言，如 Fortran、Algol、Pascal 和 C 语言等；②面向对象的编程语言，如 Simula、Smalltalk、C ++ 、Java 和 C#等。其发展历程如图 1-1 所示。

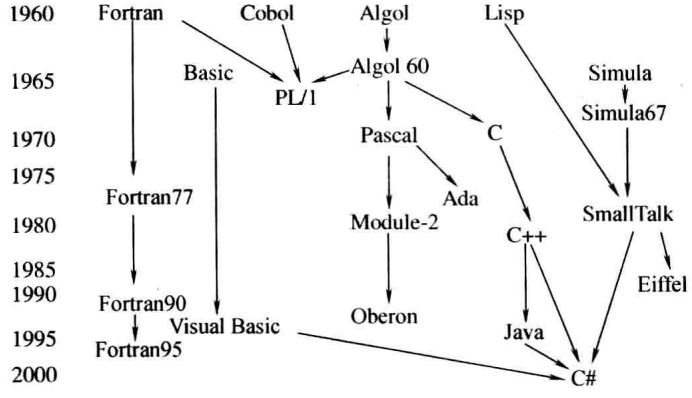


图 1-1　计算机高级语言发展历程

Fortran 语言开发于 1960 年，它是第一个高级命令语言。

Algol 语言在清晰和一致性上与以前的语言相比迈出了一大步。它是最原始的块状结构化语言并且对大部分编程语言都产生了巨大的影响。

C 语言的设计者从 Algol 语言家族继承了这一基本的思想，而且在很大程度上，C 语言还受到了为编写 UNIX 操作系统服务之意图的影响。尽管只有这些有限的原始应用，但是因为 C 语言的语法精练，运行速度快，其还是成为最流行的命令语言之一。

Pascal 语言在 20 世纪 70 年代和 80 年代非常流行，尤其在基于教学目的的学术领域中。

Fortran、Algol、Pascal 和 C 都是过程编程语言，它们都应用了面向过程的编程方式。过程语言有自己处理数据的核心过程集。每个过程包含一套按单步方式执行的指令。过程与数据在一定程度上是分离的。

Lisp 是最早的功能性语言，它广泛应用于与人工智能相关的计算编程。Lisp 与 C ++ 和 C#语言区别很大，但它包含一个被许多语言广泛接收的机制——自动内存管理。

Simula 语言产生于 20 世纪 60 年代，其是大部分面向对象语言的共同祖先。Simula 根据 Algol 60 的思想而设计，其目的是为了编写模拟程序。

Smalltalk 扩展了 Simula 的思想。它是浓缩的面向对象语言。

Visual Basic，是由微软于 1991 年开发的一种可视化的、面向对象和采用事件驱动方式的结构化高级程序设计语言，可用于开发 Windows 环境下的各类应用程序。它简单易学、效率高，且功能强大。

C ++ 在很大程度上是 C 的超级集合，它提供了大部分程序员需要的改进，如类型改进、重载子程序及面向对象编程特征。C ++ 由于继承了 C 支持操作系统开发的功能，所以是一个非常强大的语言。不幸的是，功能与自由的获取，直接与责任、复杂性以及“触及红色警报按钮”的危险性相关。换句话说，它同时产生大量的 bug。

Java 很大程度上是基于 C ++ 的一套子集而设计的，因而相对简单得多。它设计用于中性架构的构建，高移植性的程序。在执行期间，Java 虚拟机解释称做字节代码的中间语言，此中间语言与 MSIL 类似。它提供了一种类型安全引用模型、简化的继承模型、多线程及扩展的类库。

1.2 C#语言

C#语言也是一种高级编程语言，其产生于 20 世纪末。这样，人们不禁要问，同样是高级程序语言，已经存在了 C、C ++ 、Java 等高级程序语言，并且这些语言已经应用了很长时间，为何要产生 C#？到底什么是 C#语言？C#语言又有什么特点呢？它和已有的计算机语言有什么关系呢？

1.2.1 C#的来源

当微软在以 MS—DOS 主宰着全球个人计算机操作系统市场的时候，微软在软件技术上的成就，并不是很突出。微软操作系统的能力，以 MS—DOS 的 16 位单机操作系统为例，尚比不上当时 IBM 的 OS/2 或是尚未登上操作系统主流舞台的 Linux。而论开发环境，微软除了 Quick Basic 之外，大部分的开发环境仍是 Borland 的天下，Turbo C ++ 仍是当时的主流开发平台。

这个态势在微软推出 Windows 3.0 这个视窗操作系统时更为明显。虽然是自家推出的视窗环境操作系统，但具有讽刺意味的是微软并没有为这个操作系统提供任何完全视窗环境下的集成开发环境（IDE）。倒是开发软件供应商的老大哥——Borland 率先提供了 Borland C ++ 这个完全在视窗环境下就可以开发软件的开发工具。

1995 年到 1996 年，Internet 以出人意料的速度发展，同时造就了许多新兴的事业与公司。其中给人印象最深的要数 Netscape 了。Netscape 以开发 www 浏览器——Netscape Navigator 起家，在两年的时间内迅速崛起，成为 Internet 上最知名的软件公司。在同时期，Sun 也以 Java 拓展了自己的另一块事业版图。在 Sun 与 Netscape 的相互应合之下，微软这个软件帝国的霸主地位，正逐渐地受到怀疑和挑战。

1996 年，是微软反败为胜最为关键的一年。这个时候，微软终于开始正视 Internet 的威力了。在网络用户端方面，微软以免费的方式发行自家的浏览器（其实是并购来的）——Internet Explorer 来打击 Netscape。在网络服务器方面，微软在 Windows NT 4.0 中，免费附加了 IIS（Internet Information Server）2.0，内附有 WWW、FTP 以及 Gopher 等服务器，同样是希望能够以免费的手段来取得市场占有率。但最重要的技术革新，还是在 ActiveX 之前微软在 Internet 应用上所做的努力，已获得若干成效。Internet Explorer 的占有率，也终于能够与 Netscape 并驾齐驱，并且有取而代之的气势。此时，微软欠缺的就是一个全面性的网络整合平台。

1999 年初，微软推出了在 Windows NT 平台上的 Windows NT 4.0 Option Pack，其中包含了 IIS 4.0、MTS（Microsoft Transaction Server）、ASP（Active Server Pages）、信息队列（Microsoft Message Queue，MSMQ）等功能。同时，微软的第一个完全整合 Windows 与 Internet 的平台——Windows DNA 也诞生了。

在 Windows DNA 之后，微软便开始将重心转到一个新的计划——新一代的 Windows 系统（Next Generation Windows System，NGWS）。NGWS 的目的是整合新的设备和技术，希望带给用户一个全新的使用经验。NGWS 包括：Windows DNA 的新一代——Windows DNA 2000、ASP 的新一代——ASP. NET（之前名为 ASP + ）、Visual Studio 6.0 的新一代——Visual Studio. NET，以及将来的 Windows. NET（代号为 Blackcomb）。这个 NGWS 计划，就是现在所看到的. NET。C#即为其核心开发的计算机语言。

计算机编程语言（如 C ++ 和 Java）和消费类电子设备（如移动电话）的进步带来了新的问题和需求。即对各种语言编写的组件进行集成是一件很困难的事，而且由于新版本的共享组件和旧的软件不兼容，安装也常常出现问题。同时，开发人员还发现他们需要基于 Web 的应用程序，以便可以通过 Internet 进行访问和使用。移动电子设备的普及使得软件开发人员明白客户不再局限于桌面计算机系统。开发人员意识到一种软件需求：让任何人从任何地方，在任何时间使用任何设备访问 Internet 上的服务。基于这些要求，微软发布了它的 C#（读做 C Sharp）和 C#所在的集成开发环境 Visual Studio. NET（读做 dot - net，简记为 . NET）。

1.2.2 C#的定义

C#编程语言是由微软的 Anders Hejlsberg 和 Scott Wiltamuth 领导的一个小组开发的，是为 . NET 平台专门设计的语言，以便让程序员更容易转移到 . NET 平台上来。因为 C#是在 C、C ++ 和 Java 的基础上，吸取了每种语言的优点并增加了自己的特点，因此，这种转移是易于实现的。

C#主设计师 Anders Hejlsberg 给 C#下了这样一个定义："C# is a simple, modern, object oriented, and type-safe programming language derived from C and C ++. It will immediately be familiar to C and C ++ programmers. C# aims to combine the high productivity of Visual Basic and the raw power of C ++."

因此，可以给 C#总结定义为：C#是一种现代的、类型安全的、完全面向对象和可视化的编程语言，它的目标是将 Visual Basic 的高产和 C ++ 底层高效的特性结合起来。它可使得程序员能够快速而容易地为微软 . NET 平台开发解决方案。

1.2.3 C#的特点

由于C#几乎集中了所有关于软件开发和软件工程研究的最新成果，即面向对象、类型安全、组件技术、自动内存管理、跨平台异常处理等，鉴于此，C#应该是目前最好的计算机编程语言。

C#语言的特点可以归纳为以下五个方面。

1. 简单性

C#语言的简单性具体体现在：

1）C#限制了指针操作，虽仍能够使用，但在默认情况下，不允许直接对内存进行操作。在C++中所使用的操作符，如“::”和“->”都不再在C#中使用，在C#中支持的相应操作符为“.”，如果一定要对内存地址进行操作，可在不安全模式下进行。

2）因为C#是基于.NET平台的，因此，它继承了自动内存管理和垃圾回收的特点。

3）在C#中，整形数值0和1不再作为布尔值出现。C#中的布尔值是纯粹的true值和false值。“==”被用于进行比较操作而“=”被用做赋值操作。

2. 现代性

C#语言是微软为了其推出的.NET平台而开发的一门高级语言，是目前最新的计算机语言。其具体体现在：

1）C#支持支持组件编程，对于创建相互兼容的、可伸缩的、健壮的应用程序来说是非常强大和简单的。

2）C#拥有内建的支持将任何组件转换成一个web service的功能，运行在任何平台上的任何应用程序都可以通过互联网来使用这个服务。

3）C#是第一个合并XML的语言，编译器可以用其直接从源码中生成可读的文档。

3. 面向对象性

C#语言是一种真正的面向对象语言，其具体体现在：

1）与C++相比，C#没有全局变量和全局函数等，所有的代码都必须封装在类中（甚至包括入口函数Main方法），不能重写非虚拟的方法，增加了访问修饰符internal，不支持多重类继承（似Java，用多重接口实现来代替）。

2）C#中提供了装箱和拆箱机制，这样，在C#的类型系统中，每一个类型都可以看做是一个对象。

3）C#只允许单继承，即一个类只能有一个基类，从而避免了类型定义的混乱。

4. 类型安全性

C#语言具有很强的保护功能，确保使用的安全性，具体体现在：

1）取消了不安全的类型转换，如在C#中不能将double转换成boolean。

2）不允许使用未初始化的变量。值类型（常量类型）被初始化为零值而引用类型初始化时默认为null。

3）数组类型下标从零开始，进行越界检查，不让数组越界。

4）检查类型溢出。

5）用委托取代函数指针，增强了类型安全。

5. 相互兼容性

C#语言有很强的兼容性，在其集成开发环境中，各种语言可以进行交叉使用，具体体现在：

1）C#提供对 COM 和基于 Windows 的应用程序的支持。

2）允许对原始指针的有限制的使用，C#允许用户将指针作为不安全的代码段来操作老的代码。

3）用户不再需要显式地实现 unknown 和其他 COM 界面，这些功能已经内建。

4）VB. NET 和其他中间代码语言中的组件可以在 C#中直接使用。

1.2.4 C#与 Java 及 C ++

作为一种高级语言，C#语言与其他高级语言，特别是与 Java 语言又有什么异同点呢？在本小节将对 C#和 Java 以及 C ++ 之间做一比较。

C#和 Java 的相同点是非常多的，软件技术的发展同样也是一个继承和发展的过程，因此，C#和 Java 有很多相似或相同的内容是很正常的。其相同点包括：编译为机器独立、语言独立的代码，运行在托管运行环境中；采用垃圾收集机制，同时摒弃了指针；具有强有力的反射能力；没有头文件，所有代码都在包装程序集里，不存在类声明的循环依赖问题；支持多继承接口、单继承实现；所有的类都派生自 object，且必须用 new 关键字分配于堆上等。

既然 C#和 Java 之间有这么多的相同之处，那么，C#是否就是微软推出的 Java 的克隆产品？是否就是 C 与其他高级语言的简单组合呢？其实这个理解是错误的。

首先，C#不是 Java 的克隆。从实现方式上，C#与 Java 之间存在着一定的差异。Java 可以在任何有 Java 虚拟机器的平台上执行。C# 目前只能在 Windows 上执行。C# 先将代码编译为中间语言（Intermediate Language，IL），然后通过 just-in-time（JIT）的编译方式或原生码编译方式来执行；Java 可以在任何有 Java 虚拟机器（Java Virtual Machine，JVM）的平台上执行，也可以编译成原生码。而在 . NET 框架下，所有的语言都被编译为相同的 IL 代码，运行时由公共（通用）语言运行时（Common Language Runtime，CLR）负责管理，使用相同的组件，真正做到了多语言的集成。

其次，C#也不是 C 与 Java 或其他高级语言的简单组合。在设计 C#期间，考察了多种语言，如 C ++ 、Java、Modula2、C、Smalltalk 等。很多语言都包含有与 C#开发环境相同的核心思想，比如深度面向对象、简化对象等。C#和这些其他高级语言尤其是 Java 的关键不同点是它非常接近 C ++ 。C#从 C ++ 直接借用了大多数的操作符、关键字和声明。同时，C#还保留了许多被 Java 抛弃的语言特性，比如枚举等特性，在 C ++ 中，枚举显然是一个很有意义的概念，在 C#中，保留了枚举并同样使其类型安全。C#还保留了操作符重载和类型转换。C#名字空间的整体结构也非常接近 C ++ 。

另外，设计 C#语言的一个关键的目标是使其面向组件。C#加入了在编写组件时所需要的所有概念，如属性（Property）、方法、事件、特性（Attribute）和文档等。设计的特性（Attributes）是一种崭新的声明性信息，是全新的和创新的方法。它使得软件开发者不仅可以通过特性来定义设计层面的信息以及运行时信息，而且还可以利用特性建立自描述（Self-describing）组件，可为任何对象加入有类型的、可扩展的元数据。这是目前其他程序语言

所不具备的。C#也是第一个合并XML注释标记的语言，编译器可以用其直接从源码中生成可读的文档。因此，可以说C#集成了许多原有技术的优点并加以创新和提高。

总之，C++高效但是不安全，Java（跨平台）安全但是较低效，C# 安全且较高效。表1-1列出了这三种面向对象语言的功能和特点的部分比较。

表1-1 C#与C++和Java的比较

功能	C++	Java	C#
跨平台	源代码（部分）	字节码	通用语言结构CLI
执行方式	编译	编译+解释	编译+JIT转换
中间代码	无	字节码Bytecode	微软中间语言MSIL
运行环境	操作系统	JVM	CLR
内存管理	直接分配和删除	垃圾内存自动回收	垃圾内存自动回收
多重类继承	支持	不支持	不支持
操作符重载	支持	不支持	部分支持
对象访问	地址/指针	引用	引用
接口类型	无	有	有
属性成员	无	无	有
命名空间	支持	包机制	支持
指针	支持	不支持	部分非安全代码
函数指针	支持	适配器+监听程序	委托
全局函数与变量	有	无	无
无符号整数类型	有	无	有
强制类型转换	支持	不支持	支持
越界自动检查	无	有	有
多维数组	数组的数组	数组的数组	真正多维数组
索引	支持	不支持	支持
线程同步	调用函数	语言内部	语言内部
异常处理	可选	支持检查异常	只支持非检查异常
标准类库	贫乏	丰富	庞大
适用领域	面向对象的系统和界面编程	跨平台（服务器端）网络编程	基于Windows平台.NET和组件编程

表1-1中，公共语言基础结构（Common Language Infrastructure，CLI）是CLR的一个子集，为IL代码提供运行的环境，.NET环境下的任何语言编写的代码通过其特定的编译器转换为MSIL代码之后均可运行其上。