BrainySoftware

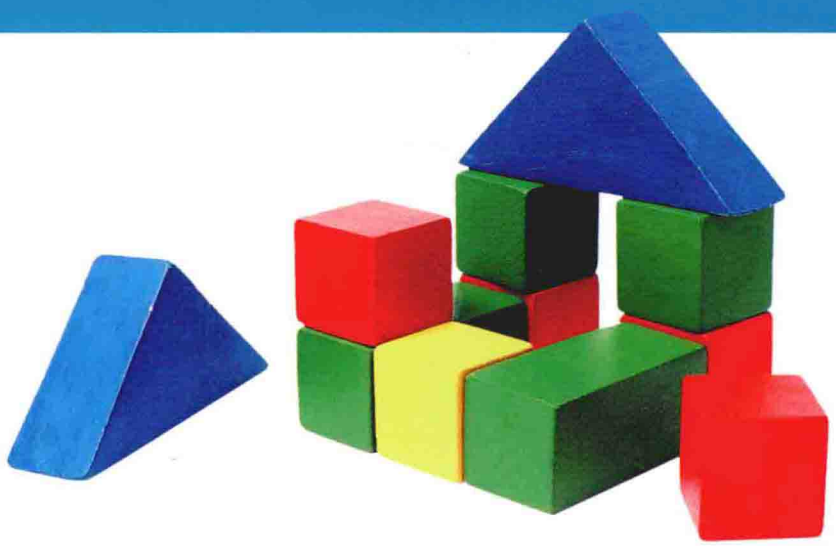
帮助初学者快速上手的编程指南

C#

初学者指南

C# A Beginner's Tutorial

[加拿大] Jayden Ky 著　　李强 吴戈 译



人民邮电出版社
POSTS & TELECOM PRESS

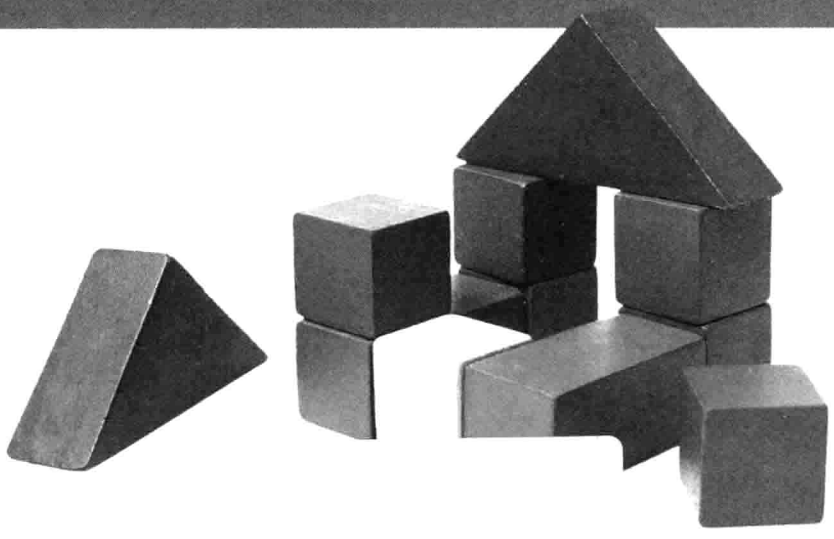
BrainySoftware

C#

初学者指南

C# A Beginner's Tutorial

[加拿大] Jayden Ky 著　　李强 吴戈 译



人民邮电出版社
北京

图书在版编目（CIP）数据

C#初学者指南 / （加）杰登（Jayden Ky）著 ；李强，吴戈译. -- 北京 ：人民邮电出版社，2014.7
ISBN 978-7-115-35290-3

Ⅰ. ①C… Ⅱ. ①杰… ②李… ③吴… Ⅲ. ①C语言－程序设计－指南 Ⅳ. ①TP312-62

中国版本图书馆CIP数据核字(2014)第073164号

内 容 提 要

C#是一种简单易学的、成熟的编程语言，作为.NET Framework 的一部分，C#语言得到非常广泛的应用。

本书是一本 C#语言的初学者的教程，涵盖了 C#和.NET Framework 语言中最重要的主题。全书共包括16 章和 3 个附录，依次介绍了 C#程序语言、面向对象编程和.NET Framework 类库 3 个方面的知识和技术。附录部分简单介绍了 Visual Studio Express 和 SQL Server Express 等常用工具。

本书内容全面，示例丰富，浅显易懂，可以帮助读者掌握 C#编程基础知识，以完成中级 C#程序员的日常任务。本书适合 C#语言初学者和对 C#编程感兴趣的读者阅读，也可以作为相关专业的教学参考书或培训教材。

◆ 著　[加拿大] Jayden Ky
译　李 强 吴 戈
责任编辑　陈冀康
责任印制　彭志环　焦志炜

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路 11 号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司

◆ 开本：800×1000　1/16
印张：15.5
字数：290 千字　2014 年 7 月第 1 版
印数：1–3 000 册　2014 年 7 月河北第 1 次印刷
著作权合同登记号　图字：01-2013-8792 号

定价：39.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315

版 权 声 明

前　言

欢迎阅读本书。

C#（读作“c sharp”）是一种易学的、成熟的编程语言。同时，它也是.NET Framework 的一部分。.NET Framework 是很大的一个技术集合，它包罗万象，以至于初学者往往不知从何入手。如果你也是一名初学者，那么本书非常适合你，因为本书就是专门为.NET 初学者所编写的教程。

作为初学者的教程，本书并不会介绍.NET Framework 中的每一种技术。相反，本书涵盖 C#和.NET Framework 语言中最重要的主题，掌握了这些内容，你才能够自学其他的技术。但是本书的内容很全面，在完全理解各章的内容后，你就能很好地完成中级 C#程序员的日常任务。

本书介绍了以下三个主题，它们是专业的 C#程序员必须要掌握的。

- C#程序语言
- C#的面向对象编程（Object-Oriented Programming，OOP）
- .NET Framework 类库

设计一门高效的 C#课程，其困难之处就在于，这三个主题实际上是彼此相关的。一方面，C#就是 OOP 语言，所以如果你已经了解 OOP，那么学习 C#的语法就较容易。另一方面，诸如继承、多态、数据封装等 OOP 的特性，我们最好是用真实案例来讲解。可是，理解真正的 C#程序，却需要我们具有.NET Framework 类库的知识。

因为这三个主题相互依赖，所以我们不能把它们划分为三个独立的部分。相反，讨论一个主题的章节会和讨论另一个主题的章节交织在一起。例如，在介绍多态之前，本书要确保我们已经熟悉某些.NET Framework 的类，以便能给出真实的案例。另外，如果不能很好地理解一组特定的类，我们就很难理解泛型这样的语言特性，而这组特定的类又是在讨论完支持类后才介绍的。

本书中也会有一个主题在两三个地方重复出现的情况。例如，for 和 while 循环语句是一个基本的语言特性，应该在前边的章节中介绍它。用 foreach 循环遍历一个数组或集合，却只能在介绍过数组和集合类型后再讲解。因此，循环语句首先会出现在第 3 章，然后在第 5 章介绍数组和第 13 章介绍集合时，会再次出现。

本前言接下来的内容会给出.NET Framework 的高级概述、OOP 的介绍、每章的简单介绍以及.NET Framework 的安装指南。

.NET Framework 概述

.NET Framework 是一种编程环境的常用名称，其规范的叫法为通用语言基础架构（Common Language Infrastructure，CLI）。CLI 是微软开发的并且通过了 ISO 和 ECMA 的认证标准。ISO 和 ECMA 都是国际标准化机构。

.NET Framework 引人注目的地方之一，就是支持多种编程语言。实际上，最新的统计结果表明，有超过 30 种语言可以使用.NET Framework，包括 Visual Basic、C#和 C++。这就意味着，如果习惯于使用 Visual Basic，可以继续用这种语言编程。如果你是一名 C++程序员，也不必为了充分利用.NET Framework 所提供的优势而去学习一种新的语言。

但是，多语言支持并不是.NET Framework 的唯一特性。它还提供了一整套技术，使软件开发更迅速且应用程序更加健壮和安全。多年来，.NET Framework 成为首选的技术，因为它具有以下优点。

- 跨语言集成
- 易用性
- 平台独立性
- 一个可以加速应用程序开发的庞大的类库
- 安全性
- 可扩展性
- 广泛的行业支持

.NET Framework 不像传统的编程环境。在传统的编程中，源代码要编译成可执行代码。这个可执行代码对于目标平台来说是本地的，因为它只能在原计划运行的平台上运行。换句话说，在 Windows 上编写和编译的代码，只能在 Windows 上运行；在 Linux 上编写的代码，只能在 Linux 上运行，依次类推，如图 I-1 所示。

图 I-1　传统的编程范式

相反，.NET Framework 程序则编译成通用中间语言（Common Intermediate Language，CIL，读作“sil”或“kil”）代码。如果你熟悉 Java，CIL 代码相当于 Java 的字节码。CIL 代码此前叫作微软中间语言（Microsoft Intermediate Language）或 MSIL 代码，只能运行在公共语言运行时（Common Language Runtime，CLR）上。CLR 是解释 CIL 代码的一个本地应用程序。因为 CLR 可用于多个平台，同样的 CIL 代码也变成了跨平台的代码。如图 I-2 所示，我们可以用支持的任何语言编写一个.NET 程序并且把它编译成 CIL 代码。同样 CIL 代码可以运行在任意已经开发了 CLR 的操作系统上。除了 CIL 代码以外，.NET 编译器还生成了元数据以描述 CIL 代码中的类型。这个元数据在术语上叫作清单（manifest）。把 CIL 代码和相应的清单一起打包成一个.dll 或.exe 文件，叫作程序集。

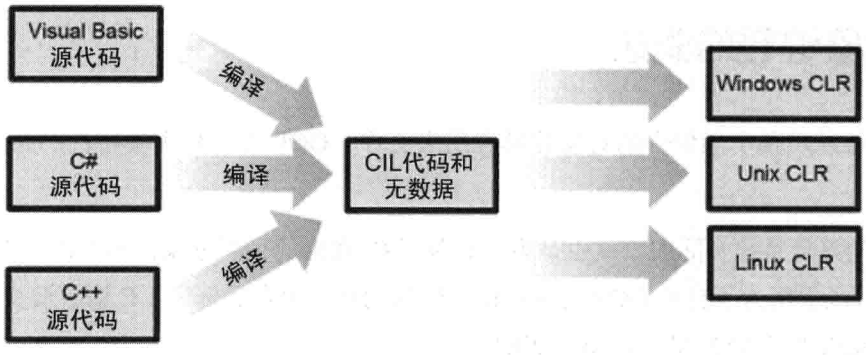


图 I-2 .NET 编程模型

目前，微软提供了 CLR 针对 Windows 的实现，但是随着来自 Project Mono（http://www.mono-project.com）和 DotGNU Portable.NET（http://dotgnu.org/pnet.html）的其他实现，CIL 代码已经能够在 Linux、Mac OS X、BSD、Sony PlayStation 3 和 Apple iPhone 上运行了。

.NET 术语中把只能在 CLR 之上运行的代码称为托管代码（Managed code）。另一方面，一些.NET 语言，诸如 C#和 C++，既可以生成托管代码又可以生成非托管代码。非托管代码运行在运行时之外。本书只介绍托管代码。

当用 C#或其他.NET 语言编程时，我们总是使用通用类型系统（Common Type System，CTS）来工作。在解释 CTS 前，我们要确定你已经了解了什么是类型。那么，什么是类型呢？在计算机编程中，类型决定了值的种类，例如一个数字或一段文本。对于编译器来说，类型信息特别有用。例如，它使得的 3*2 这个乘法运算有意义，因为 3 和 2 都是数字。但是，我们如果在 C#代码中写下 VB *C#，编译器将认为它无效，因为不能把两段文本相乘，至少，

在C#中不允许这样做。

CTS 中有 5 种类型。

- 类
- 结构
- 枚举
- 接口
- 委托

在本书中，我们会逐一介绍这些类型。

面向对象编程的概述

OOP 通过对真实世界中的对象建模应用来工作。OOP 有三种主要特性：封装、继承和多态。

OOP 的好处是很实际的。这也是为什么包括 C#在内的大部分现代编程语言都是 OO 的。我们甚至可以引用为了支持 OOP 而进行语言转换的两个著名的例子：C 语言演变成了 C++，而 Visual Basic 则升级为 Visual Basic.NET。

本节介绍 OOP 的好处并且对学习 OOP 的难易程度给出一个评估。

OOP 的优点

OOP 的优点包括代码易维护、代码可复用和可扩展性。下面，我们对这些优点作更详尽的介绍。

1. 易维护

当前的应用软件往往非常大。很久以前，一个“大”系统包含几千行代码。而现在，甚至 100 万行代码都不算是大系统。当系统变得更大的时候，就开始产生一些问题。C++之父 Bjarne Stroustrup 曾经说过类似于下面的话。小程序可以随便写，即使不是很容易，但最终你还是能让它工作。但是，大程序却截然不同。如果你没有使用“好的编程”技术，新的错误会随着你修正老错误的步伐而不断地产生。

原因就在于，一个大程序的不同部分会相互影响。当我们修改程序的某一部分内容时，可能不会意识到这种变化可能会影响到其他部分。OOP 使应用程序更容易模块化，而且模块

化使得维护不再头疼。模块化在 OOP 中是内在的，因为类（它是对象的模板）本身就是一个模块。好的设计应该允许一个类包含相似的功能和相关的数据。在 OOP 中，一个经常用到的重要的相关术语是耦合，它表示两个模块之间有一定程度的交互。各部分之间的松耦合使得代码更容易实现复用，而复用是 OOP 的另一个优点。

2．可复用性

可复用性意味着，如果对于最初编写的代码有相同的功能需求，那么代码的编写者或其他人可以复用它们。一种 OOP 语言通常带有一套现成的库，这并不奇怪。以 C#为例，这门语言是.NET Framework 的一部分，它提供了一套详细设计并经过测试的类库。编写和发布自己的库也很容易。编程平台中对可复用性的支持是非常吸引人的，因为它会缩短开发时间。

类的可复用性的最大挑战之一就是为类库创建好的文档。程序员如何才能快速地找到提供了他（或她）想要的功能的那个类？找到这样一个类会比从头编写一个新的类更快吗？好在，.NET Framework 类库有大量的文档。

可复用性不仅适用于编码阶段类或其他类型的复用，当我们在 OO 系统中设计一个应用时，OO 设计问题的解决方案也可以复用。这些解决方案叫作设计模式。为了更容易地指明每一种解决方案，人们给每种模式都起了一个名称。在经典的设计模式图书《*Design Patterns:Elements of Reusable Object-Oriented Software*》中，我们可以找到可复用的设计模式的最早的名录，该书作者是 Erich Gamma、Richard Helm、Ralph Johnson 和 John Vlissides。

3．可扩展性

每个应用都是独一无二的，都有自己的需求和规格。说到可复用性，有时我们无法找到一个已有的类能够提供应用程序所需的确切功能。但是，我们可能会找到一个或两个类，它们提供了这些功能的一部分。可扩展性意味着，我们仍然可以使用这些类，通过扩展它们来满足我们的需求。我们还节省了时间，因为不需要从头编写代码。

在 OOP 中，可扩展性通过继承来实现。我们可以扩展一个已有的类，为它添加一些方法或数据，或者修改不喜欢的方法行为。如果知道该基本功能会在许多示例中使用，但是又不希望类提供非常具体的功能，那么我们可以提供一个通用的类，以后可以扩展它来为应用程序提供具体的功能。

OOP 很难吗

C#程序员需要熟练掌握 OOP。如果你曾经使用过过程式语言，诸如 C 或 Pascal，你会发现有很大的不同。鉴于此，这既是好消息又是坏消息。

我们先说坏消息。

研究人员一直针对在学校教授 OOP 的最好的方式而争论不休。有些人认为，最好的方法是在介绍 OOP 前先教授过程式语言。我们发现，在许多学校，OOP 课程往往安排在接近大学最后 1 年。

但是，最新的研究表明，拥有过程式编程技能的人和 OO 程序员的视角以及解决问题的模式差异迥然。当熟悉过程式编程的人需要学习 OOP 时，他所面临的最大障碍就是模式的转变。据说，从过程式转变到面向对象模式，这种观念的转变需要 6～18 个月的时间。其他的研究也表明，没有学习过过程式编程的学生，不会觉得 OOP 很难。

我们再来说好消息。

C #是学习 OOP 的最简单的语言之一。例如，我们不需要担心指针，也不需要花费宝贵的时间去解决由于没有释放不用的对象而引起的内存泄漏的问题。最重要的是，.NET Framework 有一个非常广泛的类库，在其早期的版本中，bug 相对来说很少。一旦我们掌握了 OOP 的基本要点，用 C#编程是很容易的。

关于本书

本书每一章的内容可以概述如下。

第 1 章，“初识 C#”。本章编写了一个简单的 C#程序，然后用 csc 工具编译并运行它。另外，本章还给出关于编码惯例和集成开发环境的一些建议。

第 2 章，“语言基础”介绍了 C#语言的语法，还介绍了字符集、基本类型、变量和运算符等。

第 3 章，“语句”，介绍了 C#中的 for、while、do-while、if、if-else、switch、break 和 continue 等语句。

第 4 章，“对象与类”，是本书中的第一节 OOP 课程。本章通过解释什么是 C#对象以及如何在内存中存储它开始了对 OOP 的学习，然后继续介绍了类、类成员以及两个 OOP 的概念（抽象和封装）。

第 5 章，“核心类”介绍了.NET Framework 类库中重要的类：System.Object、System.String、System.Text.StringBuilder 和 System.Console，还介绍了数组。本章非常重要，因为本章所介绍

的类是.NET Framework 中最常用到的一些类。

第 6 章，“继承”，介绍了 OOP 的特性之一——继承，它使得代码可以扩展。本章介绍了如何扩展一个类、影响子类的可访问性以及覆盖方法等内容。

第 7 章，“结构”，介绍了 CTS 的第二种类型——结构。本章强调了引用类型和值类型之间的不同，介绍了.NET Framework 类库中经常用到的一些结构。本章还介绍了如何编写自己的结构。

毋庸置疑，错误处理在任何编程语言中都是一项重要特性。作为一门成熟的语言，C#有非常健壮的错误处理机制，它能防止 bug 四处蔓延。第 8 章“错误处理”详细介绍了这种机制。

第 9 章，“数字和日期”，介绍了在使用数字和日期时所要处理的三个问题：解析、格式化和操作。本章还介绍了可以帮助我们完成这些任务的.NET 类型。

第 10 章，“接口和抽象类”，解释了接口远不只是没有实现的类那么简单。接口定义了服务提供者和客户之间的一个契约。本章还介绍了如何使用接口和抽象类。

第 11 章，“枚举”，介绍了如何使用关键字 enum 来声明一个枚举类型。本章还描述了如何在 C#程序中使用枚举。

第 12 章，“泛型”，介绍了泛型。

第 13 章，“集合”，介绍了如何使用 System.Collections.Generic 命名空间的成员来组织对象和操作它们。

第 14 章，“输入和输出”，介绍了流的概念，而且介绍了如何使用流来执行输入和输出的操作。

你会发现第 15 章“WPF”的内容很有趣，因为我们将学习编写有漂亮用户界面和易用控件的桌面应用程序。

多态是 OOP 的主要支柱之一。当一个对象的类型在编译时不为人知的时候，多态是非常有用的。第 16 章“多态”介绍了这种特性并且提供了有用的示例。

访问数据库并且操作数据，这是商业应用程序中最重要的一项任务。目前有许多种不同的数据库服务器，访问不同的数据库需要不同的技能。在第 17 章“ADO.NET”中，我们介绍如何访问数据库以及操作数据库中的相关数据。

附录 A，“Visual Studio Express 2012 for Windows Desktop”，介绍了一款免费的集成开发

环境（Integrated Development Environment，IDE），它能帮助我们更有效地编写代码。Visual Studio Express 2012 for Windows Desktop 运行在 Windows 7 和 Windows 8 上，如果你使用这类操作系统，应该考虑使用它。如果你使用更早版本的 Windows，那么可以选择 Visual C# 2010 Express 作为 IDE，我们会在附录 B“Visual C# 2010 Express”中介绍它。

最后，附录 C 介绍了如何安装 SQL Server 2012 Express 这款免费的软件并创建了一个数据库。

下载和安装.NET Framework

在开始编译和运行 C#程序前，我们需要下载和安装.NET Framework 软件。

默认情况下，在.NET Framework 出现后发布的 Windows 操作系统会包含某个版本的.NET Framework 软件。Windows 7 附带的是.NET Framework 3.5。因此，如果你需要版本 4 或 4.5，那就需要单独安装它。如果你计划使用 Visual Studio，那么你很幸运，因为它已经包含了某个版本的.NET Framework，不需要再单独安装。否则，你可以通过以下链接下载 4.5 版本。

```
http://msdn.microsoft.com/en-us/library/5a4x27ek.aspx
```

要在命令行进行编译，我们需要把包含 csc.exe 文件（C#编译器）的路径添加到 PATH 环境变量中。这个路径是 C:\Windows\Microsoft.NET\Framework\v4.x.y，其中 x 和 y 是版本号。x 和 y 的实际值要根据所安装的版本来决定。例如，作者计算机上的版本是 4.0.30319。

如果你使用的是 64 位的计算机，那么路径可能如下所示：C:\Windows\Microsoft.NET\Framework64\v4.x.y。

要给 PATH 变量添加一个路径，我们首先需要用鼠标右键点击桌面上的“My Computer”图标，选择“Properties”菜单项。其次在弹出的对话框中，点击位于“Advanced”标签页或“Advanced system settings”标签页上的“Environment Variables”按钮。最后在弹出的对话框中，把上述路径添加到 System 变量列表框中当前的 Path 变量的末尾。请注意，每条路径之间必须要用分号隔开。

选择一个 IDE

IDE 是每一位程序员都要使用的工具。大多数现代 IDE 通过帮助程序员更早地找到 bug、

debug 并跟踪程序，从而显著地提高生产效率。

就.NET Framework 开发而言，它有很多 IDE 可用，但是 Microsoft Visual Studio 显然是赢家。好在，精简版的 Visual Studio，Visual Studio Express 2012 for Windows Desktop（针对 Windows 7 和 Windows8 用户）和 Visual C# 2010 Express（针对较早的 Windows 版本）是免费的。如果你还没有 IDE，那么应该现在就去下载并安装它。在第一次使用后，你还需要注册软件，以便继续使用它。

注册是免费的。

下载程序示例

本书的程序示例以及每章中问题的答案可以从以下网址下载。

`http://books.brainysoftware.com/download/csharp.zip`

首先将这个 zip 文件解压缩到一个工作路径中。现在可以开启你的 C#编程之旅了。

目 录