

.NET 开发经典名著

学习编程第一步

以零基础上手 C#开发为例

[美] Rob Miles 著
王 净 译

清华大学出版社

北 京

Authorized translation from the English language edition, entitled *Begin to Code with C#*, 9781509301157 by Rob Miles, published by Pearson Education, Inc, publishing as Microsoft Press, Copyright © 2016 by Rob Miles.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc. CHINESE SIMPLIFIED language edition published by TSINGHUA UNIVERSITY PRESS LIMITED, Copyright © 2017.

北京市版权局著作权合同登记号 图字：01-2016-9912

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

学习编程第一步：以零基础上手 C#开发为例 / (美)鲍勃·迈尔斯(Rob Miles) 著；王净 译. —北京：清华大学出版社，2017

(.NET 开发经典名著)

书名原文：Begin to Code with C#

ISBN 978-7-302-47578-1

I. ①学… II. ①鲍… ②王… III. ①C 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2017)第 142876 号

责任编辑：王 军 于 平

装帧设计：孔祥峰

责任校对：牛艳敏

责任印制：沈 露

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社 总 机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者：北京鑫丰华彩印有限公司

装 订 者：三河市溧源装订厂

经 销：全国新华书店

开 本：185mm×260mm 印 张：27 字 数：708 千字

版 次：2017 年 7 月第 1 版 印 次：2017 年 7 月第 1 次印刷

印 数：1~3000

定 价：69.80 元

产品编号：074386-01

译者序

Microsoft Visual Studio(简称 VS)是美国微软公司的开发工具包系列产品。VS 是一个基本完整的开发工具集，它包括整个软件生命周期中需要的大部分工具，如 UML 工具、代码管控工具、集成开发环境(IDE)等。所写的目标代码适用于微软支持的所有平台。C#是微软公司发布的一种面向对象的、运行于.NET Framework 之上的高级程序设计语言。C#与 Java 有着惊人的相似之处；它包括诸如单一继承、接口、与 Java 几乎同样的语法和编译成中间代码再运行的过程。但是 C#与 Java 存在明显区别，它借鉴了 Delphi 的一个特点，与 COM(Component Object Model, 组件对象模型)是直接集成的，而且它是微软公司 .NET Windows 网络框架的主角。C#是一种安全的、稳定的、简单的、优雅的、由 C 和 C++衍生出来的面向对象编程语言。它在继承 C 和 C++强大功能的同时去掉了一些它们的复杂特性(例如没有宏以及不允许多重继承)。C#综合了 VB 简单的可视化操作和 C++的高运行效率，凭借其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持，成为.NET 开发的首选语言。

本书作者 Rob Miles 是一名具有多年实践教学经验的资深教师，在英国赫尔大学从事编程教学 30 多年，同时也是一位资深的 Microsoft MVP。

全书共分为四部分，第 I 部分“编码基础”包括七章，主要介绍如何安装和使用所需的编程工具，以及 C#编程语言的基础元素。第 II 部分“高级编程”包括四章，主要介绍用来创建复杂应用程序的 C#编程语言相关功能，包括介绍如何将大型程序分解成更小的元素，如何创建用来反映待解决具体问题的自定义数据类型以及如何在存储中保存数据；第 III 部分“创建游戏”包括四章，主要介绍如何构建一些可玩的游戏，同时学习如何通过继承以及基于组件的软件设计扩展编程对象；第 IV 部分“创建应用程序”包括三章，主要学习如何创建完全成熟的应用程序，如何设计图形化用户界面，如何将程序代码与显示器上的元素连接起来以及如何构建现代应用程序。

本书图文并茂，技术新，实用性强，以大量实例对 C#功能做了详细的解释，是 C#用户不可缺少的实用参考书籍。本书可作为 C#编程人员的参考手册，适用于计算机技术人员。

本书主要由王净翻译，参与本书翻译的还有田洪、范园芳、范桢、胡训强、纪红、晏峰、余佳隼、张洁、赵翊含、何远燕、任方燕、吴同菊、曹兵、朱婷婷、蒋芬娇、王湘旭、朱荣玲、罗聪玉、戈毛毛，在此一并表示感谢。此外，还要感谢我的家人，她们总是无怨无悔地支持我的一切工作，我为有这样的家庭而感到幸福。

译者在翻译过程中，尽量保持原书的特色，并对书中出现的术语和难词难句进行了仔细推敲和研究。但毕竟有少量技术是译者在自己的研究领域中所不曾遇到过的，所以疏漏和争议之处在所难免，望广大读者提出宝贵意见。

最后，希望广大读者能多花些时间细细品味这本凝聚作者和译者大量心血的书籍，为将来的职业生涯奠定良好的基础。

王 净

作者简介



Rob Miles 在英国赫尔大学从事编程教学 30 多年。他是一名资深的 Microsoft MVP，并且热爱编程、C#以及创建新事物。只要有空闲时间，他就会钻研代码。他喜欢创建程序，然后运行并查看所发生的事情。他认为编程是可以学到的最富有创造性的技能。此外，他还认为我们与火星人之间的战争最终会以我们的胜利而告终，因为我们可以使用 Visual Studio，而火星人却不能——宇宙中没有比软件更好的东西了。

虽然他声称知道许多有趣的故事，但没有人听他讲过一个笑话。如果你想要了解 Wacky World of Rob Miles，可以在 www.robmiles.com 上阅读他的博客，并可以通过 @RobMiles 在 Twitter 上与他联系。

前言

我认为，编写程序是一项最富有创造性的活动。如果学会了画画，则可以绘制出美妙的图画。如果学会了拉小提琴，则可以奏出美妙的音乐。但如果学会了编程，则可以创造全新的体验(如果你愿意，也可以绘制出炫酷的图画以及美妙的音乐)。一旦踏上了编程这条路，那么你可以到达的目的地是没有任何限制的。通常，所学到的编程技能总有用武之地。

你可以将本书视为开启编程之旅的第一步。一旦头脑中有了奋斗的目标，那么努力的过程将是美好的，学习编程亦是如此。因此，我更愿意将学习编程的目的描述为“有用性”。虽然在学完本书后，你可能并不会成为世界上最优秀的程序员，但至少具备正确编写有用程序所需的技能和知识。至少有一个你编写的程序会在 **Microsoft Store** 中供他人下载使用。

然而，在开始学习之前，我想先给出一个小小的警示。如果你打算进行一次探险之旅，那么一定可以找到一本指南告诉你可能会遇到的狮子、老虎以及鳄鱼，同样，我认为必须让你知道的是，我们的学习旅程不可能是一帆风顺的。程序员必须学会以不同方式来思考需要解决的问题，因为计算机的工作方式与我们人类的工作方式不一样。人类可以慢慢地完成一些复杂的事情，而计算机则可以非常快速地完成一些简单的事情。程序员的工作就是如何利用计算机的一些简单的功能来解决复杂的问题。而这恰恰也是我们将要学习的内容。

成为一名合格程序员的成功要素与努力成为其他职业人的成功要素是一样的。如果想要成为一位世界知名的小提琴演奏家，就必须进行大量的练习。对于编程来说也是如此。只有花费大量的时间来编写自己的程序，才能掌握代码的编写技巧。就像小提琴演奏家真正地喜欢乐器唱歌一样，让一台计算机完成你希望的工作是一个真正令人满意的体验。当看到其他人正在使用你编写的程序并认为该程序实用且好用时，你一定会感到非常愉快。

本书的组织结构

本书共分为四个部分。每一部分都是建立在前一部分的基础之上，从而引导读者逐步成为一名成功的程序员。首先，将学习程序告诉计算机完成工作所需的低级别编程指令，然后学习一些专业的软件实践。

第 I 部分：编码基础

第 I 部分是入门，主要介绍如何安装和使用所需的编程工具，以及 **C#** 编程语言的基本元素。

第 II 部分：高级编程

第 II 部分描述用来创建复杂应用程序所使用的 C#编程语言的相关功能。该部分将介绍如何将大型程序分解成更小的元素，如何创建用来反映待解决具体问题的自定义数据类型。此外，还将学习当程序不使用某些数据时如何在存储中保存这些数据。

第 III 部分：创建游戏

创建游戏是非常有趣的。同样，事实证明，它也是学习如何使用面向对象编程技术的一种非常好的方法。在该部分，将构建一些可玩的游戏，同时学习如何通过继承以及基于组件的软件设计扩展编程对象的基础知识。

第 IV 部分：创建应用程序

在第 IV 部分，将学习如何创建完全成熟的应用程序。你将学习如何设计图形化用户界面以及如何将程序代码与显示器上的元素连接起来。学习如何构建现代应用程序。

本书的学习方法

在每一章，都会介绍一些关于编程的相关内容。首先，我会演示如何完成某个操作，然后引导读者使用所学的内容完成一些自己的操作。读者所完成的内容不会超过一个页面。每一章都会使用到 Snaps 库，该库预先创建了部分功能(本书会介绍如何使用该库)。随后，由读者来完成其他一些操作！

如果愿意，可以直接通读本书。但如果放慢阅读速度，并在学习过程中完成一些实际操作，将会学到更多内容。实际上，本书并不完全是一本教你如何编程的书，任何一本关于自行车的书都不可能教会你如何骑自行车。你必须投入大量时间和实践来学习如何编程。在你尝试进行编程时，本书可以提供相关的知识和信心，此外，当所编写的程序无法按预期运行时，本书也会为你提供帮助。本书中的所有内容将帮助你学习如何编程，努力吧！



动手实践

当然，学习的最佳方法是实践，所以你会发现本书中包含许多“动手实践”部分。该部分提供了练习编程技能的方法。首先从一个示例开始，然后介绍一些可以自己尝试完成的步骤。你创建的所有程序都会在 Windows PC、笔记本电脑或者手机上运行。甚至可以通过 Windows Store 将自己的创作成果发布到全世界。



代码分析

学习如何编程的一个好方法是研究其他人编写的代码，并弄清楚该代码完成的操作(有时还需要弄清楚为什么代码没有完成它应该完成的操作)。在本书的“代码分析”中，将使用演绎技巧来阐释一段程序的行为，讨论如何修复 bug 并提出一些改进建议。



易错点

如果你不知道程序失败的原因，那么在开始编写第一个程序之后将会很快学习到相关内容。为了帮助你提前处理相关问题，本书提供了“易错点”部分，其中预测了可能遇到的问题，并提供了相应的解决方案。例如，当介绍一些新知识时，有时我会花一些时间来考虑程序可能失败的原因以及当使用新功能时所担心的问题。

程序员要点

我曾经花费了大量时间教授如何进行编程。同时，还编写许多程序并向几个付费用户出售了一些程序。通过大量的实践，我学会了一些软件设计的方法，而这些方法我希望在你学习编程的过程中就可以掌握。“程序员要点”的目的是预先给你提供相关信息，以便你可以在学习软件开发的过程中从软件开发的专业角度思考问题。

“程序员要点”涵盖了许多问题，从编程到人再到哲学。我强烈建议你仔细阅读并吸收这些要点——在以后的工作中，这些要点可以为你节省大量时间！

程序和 Snaps 库

没有人会从头开始创建程序。所有软件都是通过使用已经构建的部分软件而构建的。如果一个程序想要显示文本，或者播放一些视频，那么只需要请求另一个程序完成相关操作即可。每种流行的计算机语言都由一个巨大的现有代码库所支持，程序员需要了解的其中一件事就是如何使用这些库以及其他别人编写的软件。

我已经为本书专门创建了 Snaps 库。该库提供了一组易用且便于组合的功能行为。在你的第一个程序中就会用到 Snaps 库。本书的后面还会介绍其他可以用来构建程序的功能库。

使用了 Snaps 库的程序需要在 Snaps 引擎中运行，Snaps 引擎是一个独立的环境，在该环境中，程序可以发出消息、从用户获取输入、绘制图像、发出声音，甚至可以知道天气的状况。

本书会提供相关的示例来说明 Snaps 库的工作原理，至于通过这些示例可以学习到什么，就由你去体会了。我们所遵循的原则是“如果你无法通过编程让你的朋友和家人留下深刻印象，那么该程序的意义是什么呢？”我真的希望你可以构建一些令人印象深刻的程序，甚至可以发布给其他人使用。

程序员要点

一切都建立在别人的代码之上

第一个程序员要点是，一名优秀的程序员可以“创造性地偷懒”，这看起来是非常合理的。如果可以使用已编写好的程序，就没有必要编写类似的程序。本书提供的 Snaps 库就是这样一个示例。本书的后面会介绍该库的一些内部代码，你会发现，它本身就使用了其他库。

软件和硬件

如果想要运行本书中的程序，需要一台计算机以及一些软件。我想我恐怕无法为你提供一台计算机，但在本章，你会找到在哪里可以获取 Visual Studio 2015 Community Edition，可以使用这个免费的软件来创建自己的程序。此外，还会知道到哪里下载 Snaps 库以及用来学习和使用的演示代码。

计算机必须运行 Windows 10 操作系统的 64 位版本，此外，还要满足以下需求：

- 1GHz 或者更快的处理器，最好是 Intel i5 或更高版本。
- 至少 4GB 的 RAM，但最好是 8GB 或以上。
- 完整的 Visual Studio 2015 Community 安装需要占用 8GB 的磁盘空间。

对于图形显示器没有具体的要求，但是当编写代码时，高分辨率的屏幕可能让你看到更多内容。Snaps 库可以使用触摸屏、鼠标、笔输入设备以及 Xbox One 和 Xbox 360 控制器(针对第III部分开发的游戏)。

Visual Studio 2015 Community Edition 是一款免费的应用程序，可用来在 Windows 10 PC 上创建 C#程序。如果你的计算机上只安装了 Visual Studio 的更早版本(比如 Visual Studio 2013)，那么恐怕无法使用该版本来完成书中的程序。然而，Visual Studio 的 2015 版本可以与现有的安装一起使用。在第 1 章，我会提供一个链接来详细指导如何安装和运行 Visual Studio。但为了更好地使用 Visual Studio，你最好拥有一个 Microsoft 账号，以便为你分配一个开发许可证。

下载

在本书的每一章，我都会演示并解释一些程序，从而教你如何开始编程——当然，你也可以使用程序来创建自己的程序。可以下载 Snaps 库、书中的示例代码、Visual Studio 的安装和设置说明(请从以下页面下载)：

<https://aka.ms/BeginCodeCSharp/downloads>

按照第 1 章以及安装文档的说明就可以成功安装示例程序和代码。也可扫描封底的二维码获取下载资源。

致谢

我真的非常喜欢撰写图书。首先，非常感谢 Microsoft Press 的 Devon Musgrave 和其他工作人员，感谢他们给予我撰写本书的机会，其次感谢 Rob Nance 所提供的美妙艺术品以及 John Pierce 和 Lance McCarthy 在文本方面所完成的奇妙工作。事实证明，他们所给予的帮助是非常重要的，正是由于他们的帮助，才确保了本书的所有内容尽量正确无误。

勘误表、更新和图书支持

我们已经尽最大的努力来确保本书及其相关内容的正确性。可以通过以下页面访问本书的更新——其形式为勘误表以及相关修改的列表：

如果你发现了未包含在该列表中的错误，请通过相同的页面将该错误提交给我们。

如果需要额外的支持，请发送电子邮件到 Microsoft Press Book Support(mspinput@microsoft.com)。

注意，上面的地址并不会提供 Microsoft 软件和硬件的产品支持。如果想要得到 Microsoft 软件或硬件的帮助，请访问 <http://support.microsoft.com>。

我们想要聆听你的想法

对于 Microsoft Press 来说，你的满意是我们的重中之重，而你的反馈则是我们最大的财富。请告诉我们你关于本书的想法：

<http://aka.ms/tellpress>

我们知道你肯定非常忙，所以只提供了几个简短的小问题。你的答案会直接发送给 Microsoft Press 的编辑们(并不需要提供任何个人信息)。在此先对你表示感谢！

保持联系

让我们在 Twitter 上保持联系：<http://twitter.com/MicrosoftPress>。

目 录

第 I 部分 编程基础	
第 1 章 入门	3
1.1 建立工作场所	3
1.2 获取工具和示例	4
1.3 使用工具	4
1.3.1 Visual Studio 项目和 解决方案	5
1.3.2 通过 Visual Studio 运行程序	6
1.3.3 在 Visual Studio 中停止 程序运行	8
1.3.4 MyProgram 应用程序	8
1.4 所学到的内容	12
第 2 章 什么是编程	13
2.1 如何成为一名程序员	13
2.1.1 编程和聚会策划	13
2.1.2 编程和问题	14
2.1.3 程序员和人们	15
2.2 将计算机作为数据 处理器使用	16
2.2.1 机器、计算机和我们	16
2.2.2 使程序工作	18
2.2.3 将程序作为数据 处理器使用	19
2.3 数据和信息	25
2.4 所学到的内容	27
第 3 章 编写程序	29
3.1 C# 程序结构	29
3.1.1 确定资源	30

3.1.2 开始类定义	31
3.1.3 声明 StartProgram 方法	31
3.1.4 设置标题并显示一条消息	32
3.2 额外 Snaps	34
3.3 创建新程序文件	35
3.4 额外 Snaps	41
3.4.1 Delay	41
3.4.2 SetTextColor	42
3.4.3 SetTitleColor	42
3.4.4 SetBackgroundColor	43
3.5 创建自己的颜色	43
3.6 所学到的内容	45
第 4 章 在程序中使用数据	47
4.1 变量	47
4.1.1 变量和计算机存储	48
4.1.2 声明一个变量	48
4.1.3 简单的赋值语句	49
4.2 在程序中使用变量	50
4.2.1 在声明中分配值	52
4.2.2 将字符串添加到一起	52
4.3 使用数字	54
4.3.1 整数和实数	54
4.3.2 执行计算	56
4.4 使用不同的数据类型	58
4.5 程序中的整数和实数	60
4.5.1 变量类型和表达式	61
4.5.2 精度和准确度	62
4.5.3 通过强制转换实现 类型转换	63

4.5.4 在表达式的操作数上 使用强制转换	63	6.2 使用 while 循环执行 输入验证	101
4.5.5 类型和错误	64	6.3 使用 Visual Studio 跟踪 程序的执行	102
4.6 额外 Snaps	65	6.3.1 添加断点	103
4.6.1 天气 Snaps	65	6.3.2 命中断点	104
4.6.2 ThrowDice	65	6.3.3 查看程序中变量的内容	104
4.7 所学到的内容	66	6.3.4 单步调试程序语句	104
第 5 章 在程序中作决策	67	6.3.5 程序继续	105
5.1 理解 Boolean 类型	67	6.4 通过在循环中计数来创建 一个乘法表辅助程序	106
5.1.1 声明一个 Boolean 变量	68	6.5 使用 for 循环结构	109
5.1.2 Boolean 表达式	68	6.6 跳出循环	111
5.2 使用 if 结构和运算符	69	6.7 使用 continue 返回到循环 顶部	113
5.2.1 关系运算符	71	6.8 额外的 Snaps	115
5.2.2 等式运算符	71	6.8.1 声音输入	115
5.2.3 比较字符串	72	6.8.2 机密数据录入	116
5.3 创建语句块	74	6.9 所学到的内容	116
5.4 使用逻辑运算符创建 复杂条件	76	第 7 章 使用数组	119
5.5 添加注释使程序更清楚	78	7.1 冰淇淋销售程序	119
5.6 游乐场的游乐设施和程序	80	7.2 生成数组	121
5.6.1 读取数字	82	7.2.1 使用索引	122
5.6.2 使用 if 条件构建逻辑	83	7.2.2 使用数组	123
5.6.3 完成程序	84	7.2.3 使用 for 循环显示数组的 内容	127
5.7 使用程序资产	85	7.2.4 显示用户菜单	128
5.7.1 Visual Studio 中的资产 管理	86	7.2.5 使用冒泡排序对数组 进行排序	129
5.7.2 播放声音资产	86	7.2.6 找到最大和最小的 销售值	135
5.7.3 显示图像内容	87	7.2.7 计算出总销售额和平均 销售额	136
5.8 所学到的内容	89	7.2.8 完成程序	137
第 6 章 使用循环重复操作	91	7.3 多维数组	138
6.1 使用循环做一个 Pizza Picker	91		
6.1.1 计算选择	92		
6.1.2 显示总数	94		
6.1.3 获取用户选项	94		
6.1.4 添加一个 while 循环	96		

7.3.1 使用嵌套 for 循环处理 二维数组	139	9.3 使用 Snaps 创建一个绘 图程序	187
7.3.2 创建程序的测试版本	141	9.3.1 在屏幕上绘制点	188
7.3.3 计算数组维度的长度	141	9.3.2 使用 DrawDot Snap 在 屏幕上绘制一个点	189
7.4 使用数组作为查询表	143	9.3.3 SnapsCoordinate 结构	190
7.5 所学到的内容	144	9.3.4 使用 GetDraggedCoordinate Snap 检测绘制位置	192
第 II 部分 高级编程		9.3.5 使用 SetDrawingColor Snap 设置绘制颜色	193
第 8 章 使用方法简化编程	149	9.3.6 使用 ClearGraphics Snap 清除屏幕	194
8.1 如何创建方法	149	9.3.7 SnapsColor 结构	195
8.1.1 向类添加方法	150	9.4 创建枚举类型	195
8.1.2 通过使用参数向方法 提供信息	152	9.5 使用 switch 结构进行决策	197
8.1.3 从方法调用返回值	155	9.6 额外 Snaps	199
8.2 创建一个小型联系人 应用程序	157	9.6.1 GetTappedCoordinate	199
8.2.1 读取联系人详细信息	159	9.6.2 DrawLine	199
8.2.2 存储联系人信息	160	9.6.3 GetScreenSize	200
8.2.3 使用 Windows 本地存储	160	9.6.4 PickImage	201
8.2.4 使用引用参数传递方法 调用的结果	162	9.7 所学到的内容	201
8.2.5 显示联系人详细信息	166	第 10 章 类和引用	203
8.3 向方法中添加智能感应 注释	169	10.1 创建 Time Tracker	203
8.4 所学到的内容	171	10.1.1 创建一个用来保存 联系人信息的结构	203
第 9 章 创建结构化数据类型	173	10.1.2 当使用对象时用 this 进行引用	205
9.1 使用结构存储音符	173	10.1.3 管理多个联系人	207
9.1.1 创建和声明结构	175	10.1.4 创建测试数据	208
9.1.2 创建结构值的数组	176	10.1.5 设计 Time Tracker 的 用户界面	209
9.1.3 结构和方法	177	10.1.6 构建 Time Tracker 程序	209
9.1.4 构建结构值	179	10.1.7 创建新联系人	211
9.1.5 创建一个音乐录音机	182	10.1.8 查找客户详细信息	211
9.1.6 创建预设数组	184	10.1.9 向联系人添加分钟数	213
9.2 对象和责任: 让 SongNote 播放自己	185		

10.1.10	显示摘要	214	11.4.3	LoadGraphicsPNGImage- FromLocalStore 方法	255
10.2	结构和类	216	11.4.4	DateTime 结构	255
10.2.1	排序和结构	216	11.4.5	获取当前日期和时间	256
10.2.2	排序和引用	217	11.4.6	渐变日期和时间显示	256
10.2.3	引用和值类型	218	11.4.7	使用日期和时间创建 文件名称	257
10.2.4	引用和赋值	219	11.4.8	创建 Drawing 类	257
10.2.5	类和构造函数	223	11.4.9	创建图形列表	259
10.2.6	类引用的数组	224	11.4.10	创建绘图日记方法	260
10.3	从数组到列表	225	11.5	所学到的内容	262
10.3.1	遍历数据列表	227	第 III 部分 创建游戏		
10.3.2	列表和索引值	228	第 12 章	使用什么创建游戏	267
10.3.3	结构列表	228	12.1	创建视频游戏	267
10.4	使用 JSON 存储数据	228	12.1.1	游戏和游戏引擎	267
10.4.1	Newtonsoft JSON 库	229	12.1.2	游戏和精灵	269
10.4.2	存储和恢复列表	230	12.2	所学到的内容	280
10.5	使用 XML 获取数据	233	第 13 章	创建游戏	281
10.6	所学到的内容	236	13.1	创建一个玩家控制的球拍	281
第 11 章	使用对象构建解决方案	239	13.2	向游戏添加声音	285
11.1	创建完整的对象	239	13.3	在游戏中显示文本	286
11.1.1	保护对象中保存的 数据	239	13.4	所学到的内容	294
11.1.2	为私有数据提供 Get 和 Set 方法	242	第 14 章	游戏和对象的层次结构	297
11.1.3	提供反映对象 使用的方法	243	14.1	游戏和对象: Space Rockets in Space	297
11.2	使用属性管理数据访问	245	14.1.1	构建一个可以移动的 星星精灵	298
11.3	管理对象构建过程	249	14.1.2	允许方法被重写	304
11.3.1	捕获并处理异常	250	14.1.3	创建一个移动的星空	304
11.3.2	创建用户友好的 应用程序	252	14.1.4	基于 MovingSprite 创建 火箭	306
11.4	将图形保存在文件中	253	14.1.5	添加一些外星人	308
11.4.1	SaveGraphicsImage- ToFileAsPNG	253	14.2	设计类层次结构	313
11.4.2	SaveGraphicsImageToLocal- StoreAsPNG 方法	254	14.3	所学到的内容	315

第 15 章 游戏和软件组件	317	第 17 章 应用程序和对象	365
15.1 游戏和对象	317	17.1 创建一个计算测验程序	365
15.1.1 创建协同操作的对象	318	17.1.1 对象和用户显示	366
15.1.2 对象和状态	324	17.1.2 创建一个测验对象	367
15.1.3 接口和组件	330	17.1.3 创建测验显示页面	371
15.2 所学到的内容	334	17.1.4 添加声音和图片	373
第IV部分 创建应用程序		17.2 支持多个测验	376
第 16 章 使用对象创建用户界面	339	17.3 所学到的内容	381
16.1 创建一个加法机	339	第 18 章 高级应用程序	383
16.1.1 使用 XAML 创建一个 图形用户界面	340	18.1 提高 C#代码编写速度	383
16.1.2 XAML 元素和软件 对象	342	18.2 创建一个 Windows 10 联系人编辑器	385
16.2 创建一个新应用程序	344	18.2.1 存储联系人详细信息	386
16.2.1 创建一个空程序	346	18.2.2 存储多个联系人	388
16.2.2 使用 XAML 创建用户 界面	348	18.2.3 创建测试联系人	389
16.2.3 预览 XAML 屏幕显示 尺寸	351	18.2.4 查找联系人	390
16.2.4 添加程序行为	352	18.2.5 显示找到的联系人 列表	391
16.2.5 计算结果	354	18.2.6 编辑联系人	394
16.2.6 事件和程序	355	18.2.7 用户界面中的数据 绑定	398
16.2.7 使用 TextBox 属性 改进用户界面	359	18.2.8 使对象可观测	399
16.3 所学到的内容	362	18.3 软件设计和 Time Tracker	404
		18.4 所学到的内容	412



第 I 部分

编程基础

让我们开始编程的启蒙阶段。首先，学习安装所需的编程工具。然后你会了解计算机实际可以完成的工作以及什么是编程语言。此外，还会首次尝试使用 C# 语言来告诉计算机完成所需的工作，并且了解如何使用 Snaps 库(为了在后面的程序中使用而创建的辅助程序)。

本部分旨在介绍本书中所有程序使用的 C# 编程语言的基本要素，而在第 II 部分，将学习建立在这些编程基础之上的现代编程语言(比如 C#)，以便更容易地创建应用程序。

第 1 章

入 门

本章主要内容:

当程序员创建应用程序时, 需要使用一组工具和相关技术。在本章, 我们将学习编写程序需要使用哪种类型的计算机以及如何找到并安装生成书中代码所需使用的工具。此外, 还会通过使用书中的 C# 示例应用程序迈出实际编码的第一步。

- 建立工作场所
- 获取工具和演示
- 使用工具
- 所学到的内容

1.1 建立工作场所

如果你是一名花费了大量时间在不同国家之间运输货物的卡车司机, 那么一定希望拥有一个舒适座椅的卡车、良好的道路视野以及轻松的驾驶控制。此外, 如果卡车拥有强劲的动力能够以合适的速度爬山, 并可以轻松地应对弯曲的山腰道路, 就更好了。

同样的道理, 如果需要花费大量的时间在键盘上编写程序, 那么拥有一个像样的工作场所是很有必要的。如果愿意, 只需在某处安装一台计算机、一个键盘以及一个屏幕, 然后搬来一把椅子就可以开始工作了(如果你不介意长时间坐在该椅子上工作)。

虽然编写程序并不一定需要非常好的计算机, 但所使用的计算机应该拥有足够的内存和处理器性能来处理所使用的工具。我建议应该使用至少带有 i5 或相当的处理器、4GB 内存以及 256GB 磁盘空间的 Windows 10 设备。虽然也可以使用速度稍慢的处理器, 但这些处理器会在一定程度上阻碍开发进程, 因为在你对程序进行任何更改后, 这些处理器可能会花费一段时间来更新程序。

还有一件非常重要的事情是必须拥有一台运行 Windows 10 的 64 位版本的计算机。目前, 已经有很少的设备运行 Windows 10 的 32 位版本。虽然 32 位版本对于大多数应用程序来说都是适用的, 但在该版本中却无法使用 Visual Studio 来构建 Windows 10 应用程序。

1.2 获取工具和示例

我们将要使用的所有工具都是免费下载和安装的。此类强大的软件对任何人都是免费使用的事实让人感到惊叹。**Visual Studio** 程序使创建应用程序和游戏变得非常简单。它甚至可以帮助你应用程序放在 **Windows Store** 上售卖。

我强烈建议那些开始学习编程的人至少将自己的一个程序放到市场上。一旦想到自己编写的软件在市场上供其他人使用，是一件非常有趣的事情。

然而，在开始共享或出售编程产品之前，首先必须下载和安装编程所需的工具。根据网络连接速度的不同，安装所需要的时间也不相同。有时，需要坐下来等待从 **Internet** 上获取相关工具并进行安装。当 **Visual Studio** 下载并安装完毕之后，可能还需要花费一些时间来整理并完成一些相关操作。注意：按照我提供的顺序执行相关操作是非常重要的，但只需要在每一台计算机上完成一次安装即可。

安装 **Visual Studio** 所遵循的步骤有时可能会因为是否拥有一个 **Microsoft** 账户或者因为其他原因而有所变化。因此，在本章中，我并没有介绍详细的步骤，而是在网上提供了相关的信息，以便可以在需要的时候进行更新。

如果你还没有下载本书前言中“下载”部分所介绍的示例代码以及其他在线内容，那么可以访问下面的网站并下载文件：

<https://aka.ms/BeginCodeCSharp/downloads>

打开名为 **GettingStarted.pdf** 的文件，按照文件中提供的操作指南安装 **Visual Studio Community 2015** 并提取和安装示例代码和应用程序。

安装完成只是做好了开始的准备，请使用 **File Explorer** 打开示例代码中的文件夹。此时，应该可以看到如图 1-1 所示的文件和文件夹。现在，可以打开 **Visual Studio**，并使用 **C#** 进行编码(下一节将学习如何编码！)。

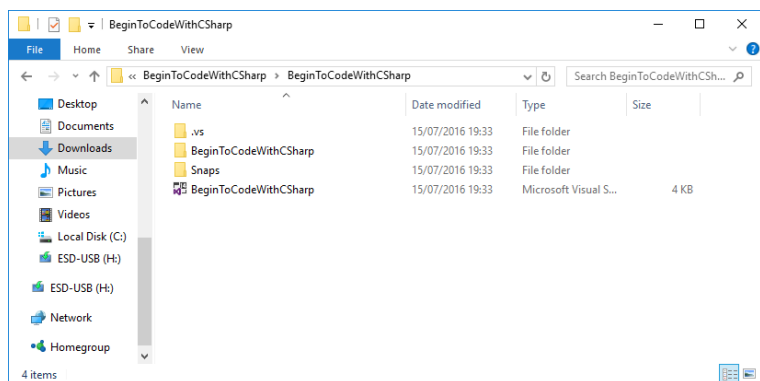


图 1-1 **BeginToCodeWithCSharp** 文件夹中的内容。稍后将使用此内容开始学习之旅

1.3 使用工具

我们已经迈开了学习如何编程的第一步。现在，可以打开 **Visual Studio** 并开始使用本书的

演示代码了。这就好比是打开一个新公寓或房子的前门，或者得到一辆闪亮的新车。

1.3.1 Visual Studio 项目和解决方案

如本节所述，Visual Studio 以项目(project)或解决方案(solution)的形式组织编程工作。当使用 Visual Studio 开发应用、应用程序、网站、Web App、脚本、插件或者其他程序时，都会创建一个新项目。项目包含了开发程序所需要使用的资源(代码文件、图像等)。当创建项目时，Visual Studio 还会创建一个解决方案，并在该解决方案中包含项目。一般来说，一个解决方案包含单个项目，但是当所开发的程序需要使用其他项目所包含的资源时，可向解决方案中添加额外项目。解决方案中的所有项目都由 Visual Studio 进行合并，从而使解决方案正常工作。

就像字处理程序可以与某一文档文件相关联一样，Visual Studio 也可以自动与解决方案文件(.sln 文件)相关联。这意味着当打开解决方案 BeginToCodeWithCSharp 时(如图 1-2 所示)，会自动打开 Visual Studio 环境以及该解决方案。现在，请找到该解决方案文件并打开(例如，双击该文件)。

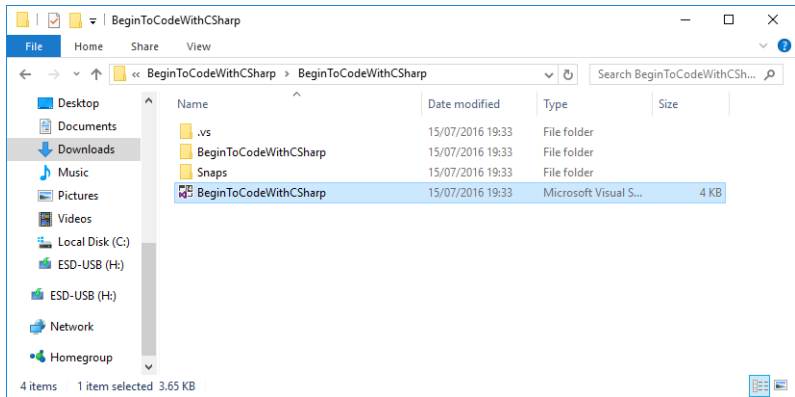


图 1-2 通过双击解决方案文件打开对应的解决方案

Visual Studio 是一种保护性程序，不会自动相信任何从 Internet 上下载的项目，所以它会询问是否确认该解决方案中的项目是可信任的，如图 1-3 所示。此时，相关项目都是可信任的(毕竟这些代码都是我编写的)，所以请选择 OK。

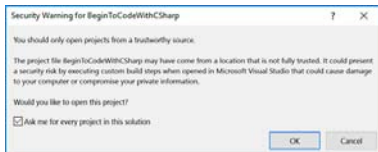


图 1-3 Visual Studio 确定是否信任所选择打开的文件

Visual Studio 提供了一个名为 Solution Explorer 的工具，可以使用该工具浏览解决方案中的项目，以及查看解决方案和项目中的每一个文件。Solution Explorer 提供了一个解决方案和项目的有组织的视图，所以让我们看一下该视图。图 1-4 显示了 Visual Studio 打开 BeginToCodeWithCSharp 解决方案时所看到的内容(根据所安装的选项不同，你所看到的内容可能也略有不同)。

该解决方案包含了两个项目，分别为 BeginToCodeWithCSharp 项目和 Snaps 项目，该项目包含了示例应用程序中使用的一组工具。Snaps 项目提供了可以被任何程序使用的基础功能，

当然也包括稍后将会创建的程序。本书后半部分会详细介绍 Snaps 的相关内容。

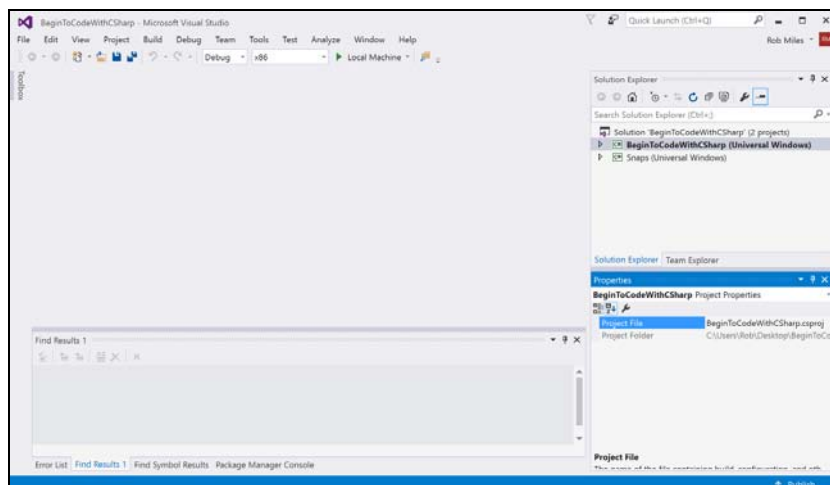


图 1-4 Visual Studio 的主页面。Solution Explorer 位于该页面的右上角

1.3.2 通过 Visual Studio 运行程序

Visual Studio 也被称为 IDE(Integrated Development Environment, 集成开发环境)。在该环境中, 程序员不仅可以编写自己的程序代码, 还可以查看程序的运行。下面运行一些由书中示例代码所创建的示例应用程序, 以此进一步了解 Visual Studio。

如果想要控制一个程序, 可以使用 Run 按钮来告诉 Visual Studio 运行程序。该 Run 按钮就是图 1-5 中指出的绿色箭头。

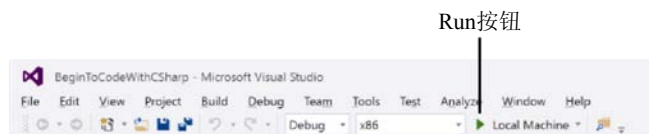


图 1-5 使用 Visual Studio 的 Run 按钮启动程序

当使用 Run 按钮启动一个应用程序时, Visual Studio 实际完成了两件事情。首先, 通过使用解决方案中管理的组件创建该应用程序。该过程被称为构建应用程序。应用程序构建完毕之后, Visual Studio 交出控制权并运行程序。对于本书的解决方案, 构建过程需要使用一个 Internet 连接才能完成。

接下来单击 Run 按钮, 运行应用程序。此时, Visual Studio 将显示如图 1-6 所示的“Begin to Code with C#”窗口。如果愿意, 可以在屏幕周围移动窗口, 最小化窗口, 或者使用右上角的 Maximize 按钮(正方形图标)全屏显示窗口。

解决方案 BeginToCodeWithCSharp 是为了让你浏览本书的示例应用程序而创建的一个应用程序。换句话说, 书中的每个演示程序都作为一个单独的应用程序包含在该解决方案中, 通过运行该解决方案, 可以选择想要运行的特定示例应用程序。其中某些程序是可以直接使用的完全成熟的应用程序, 而另一些程序则是为了便于学习而创建的特定编程知识点的简单演示。

窗口底部的按钮是用来控制这些示例应用程序运行的。如果想要运行某一应用程序, 首先

需要使用按钮上方的导航面板选择该程序。在左边名为 **Folder** 的面板中，可以选择某一特定的应用程序文件夹，大多数文件夹都以本书章节的名称命名。而在右侧的面板中，可以选择对应章节中的某一应用程序。最后，当单击 **Run an app** 按钮时，就会运行所选择的应用程序。

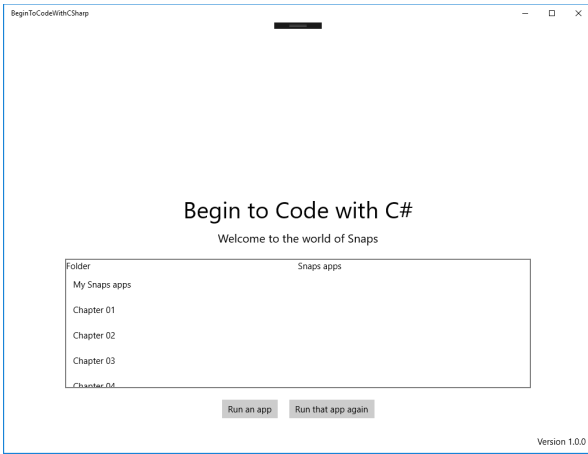


图 1-6 正在运行的 BeginToCodeWithCSharp 应用程序

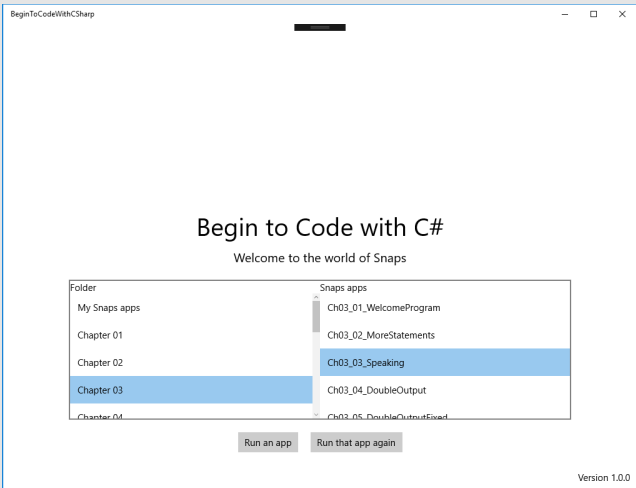


动手实践

选择并运行一个应用程序

这是我们第一个“动手实践”侧边栏。在该侧边栏中，有时我会要求你完成一些事情，有时会要求你简单地尝试一下相关操作。但不管怎样，你所做的都是开发人员可能完成的操作。此时，我仅要求你选择并运行 **BeginToCodeWithCSharp** 文件夹中的部分应用程序(当然，这也是本书中最简单的“动手实践”)。

请确保解决方案可以正常运行。首先从左边的面板中选择一个文件夹，然后从右边的面板中选择一个示例应用程序。此时有许多应用程序可供选择。可以先看一下 **Chapter03** 文件夹中的 **Ch_03_03_Speaking** 应用程序，该程序可以让计算机通过声音的方式介绍自己。



单击 **Run an app** 按钮，运行所选择的应用程序。当关闭应用程序时，导航面板会再次显示。选择并运行更多的应用程序。如果你喜欢玩游戏，可以运行 **Chapter13** 文件夹中的 **Ch_13_08_KeepUpGame**。

1.3.3 在 Visual Studio 中停止程序运行

在尝试完一些应用程序之后，需要停止 **BeginToCodeWithCSharp** 应用程序。之所以如此，是因为当运行该程序时 **Visual Studio** 将会禁止对程序的内容进行任何更改。(就好像飞机仍然在空中飞行时无法对其进行维修一样。)当然，一旦停止运行，就可以对程序进行修改了。

如果想要停止程序，只需要使用右上角的 **Close** 按钮(X)关闭窗口即可，对于任何其他的应用程序也可以执行相同的操作。然而，除此之外，**Visual Studio** 还提供了一个可用来停止运行程序的 **Stop** 按钮(如图 1-7 所示)。当你的程序因为某种原因而“卡”住时，可以使用该按钮关闭程序。

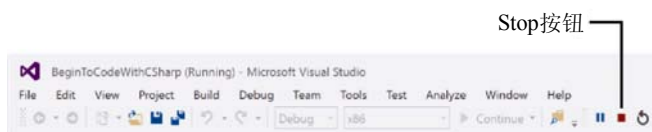


图 1-7 使用 Visual Studio 中的 **Stop** 按钮停止运行程序

重复运行应用程序

可以使用 **Run that app again** 按钮再次运行最后运行的应用程序。即使日后关闭了计算机然后再返回到 **BeginToCodeWithCSharp** 解决方案，最后运行的应用程序的名称也会被记住。



易错点

应用程序被“卡住”

欢迎来到第一个“易错点”侧边栏。在该侧边栏中，我们会了解一些在编写代码时可能遇到的陷阱，同时还会考虑如何在示例应用程序之间切换。

一些应用程序只是简单的演示，运行完毕之后就可以关闭，可以通过主应用程序的导航面板选择并运行。而另一些应用程序则需要持续运行，就像一个“真正的”应用程序。例如，你可能已经发现，无法停止 **Keep Up!** 游戏；该程序被设计为持续运行。

如果你发现自己“卡在”某一应用程序中，并且想要运行其他的程序，那么可以从 **Visual Studio** 中停止 **BeginToCodeWithCSharp** 解决方案(具体方法如前所示)或者选择运行程序右上角的 **X** 按钮停止。虽然前面已经详细介绍了如何停止一个程序，但在此处包括这么一个“易错点”侧边栏的目的是提醒你在学习的过程中应该注意的问题。

1.3.4 MyProgram 应用程序

如你所见，当首次运行 **BeginToCodeWithCSharp** 解决方案时，应用程序会显示一个欢迎消息“**Welcome to the world of Snaps**”。当然，该消息也可以由内置于解决方案的一个程序来显示。下面查看一下完成显示的 C# 代码。

在 Visual Studio 中,管理所编写程序的方法与 Windows 操作系统中管理文件的方式是相同的。当创建了一个程序时,通常会生成许多不同的部分,如图 1-8 所示。例如,现代应用程序通常会包含声音和图像,所有这些项需要保存在一起,以便用来构建已完成的程序。

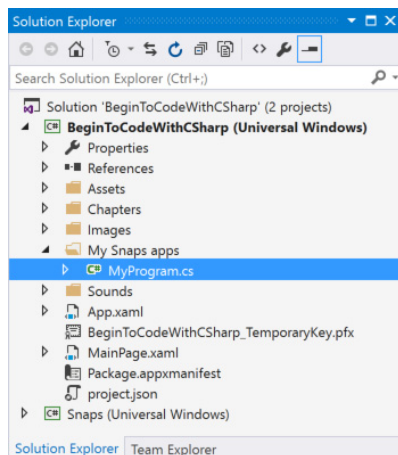


图 1-8 Solution Explorer 中的 MyProgram.cs 源文件和其他程序资源

在 Visual Studio 中, Solution Explorer 可帮助程序员管理程序的不同元素。可将 Solution Explorer 视为一种特殊的文件浏览器。它提供了一个文件夹视图,其中包含了在 Visual Studio 中构建和运行应用程序所使用的所有文件。BeginToCodeWithCSharp 项目包含了多个保存不同文件的文件夹。可以通过单击项附近的箭头来导航解决方案中的元素和文件夹。在本书后面,我们将学习更多相关的文件夹,但目前,只需要关注 MyProgram.cs 源文件(存储在 My Snaps apps 文件夹中,如图 1-8 所示)即可。该文件包含了 BeginToCodeWithCSharp 解决方案启动时所运行的程序代码。

如果在 Solution Explorer 中双击该文件,就会在 Visual Studio 的编辑器窗口中显示其代码,如图 1-9 所示。

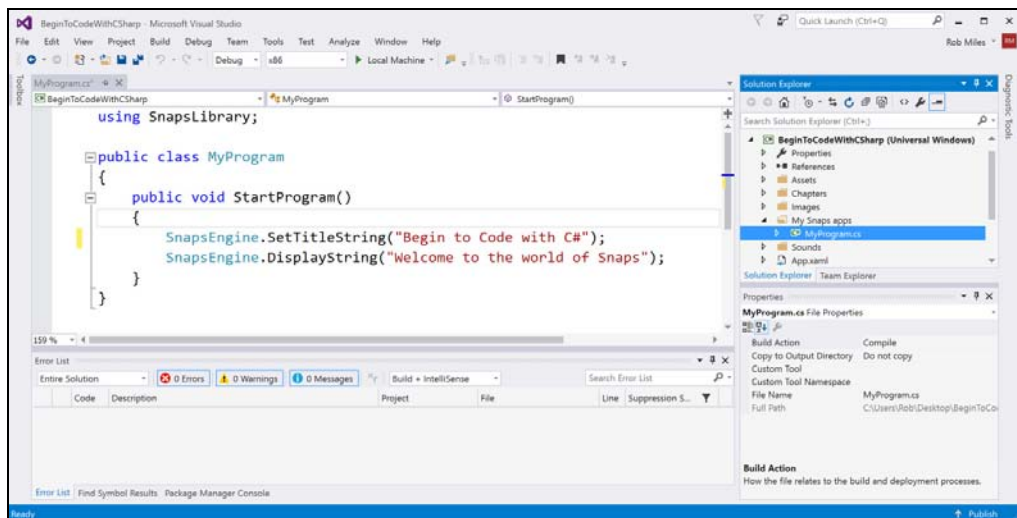


图 1-9 Visual Studio 在编辑器窗口中显示 MyProgram.cs 文件的内容

虽然所完成的操作与在一个字处理程序中打开一个文档是相似的，但所看到的并不是一连串构成某一故事(或者诗歌、报告)的单词，而是计算机运行程序时所执行的一系列指令。换句话说，你正在检查程序的代码。图 1-9 中所示的内容就是实际的 C#代码。所以，恭喜你！你已经第一次看到了 C#代码片段。

程序员要点

编程语言并不是那么特别

如果将编程语言想象为非常复杂且难以理解的事情，那么就大错特错了。我想大部分的人(当然主要指那些会英语的人)都可以理解图 1-9 中用来显示消息“Begin to Code with C#”和“Welcome to the world of Snaps”的程序。

虽然程序中的框架部分来自 C#语言(主要是为了便于理解)，但其他的部分都精心使用了用来描述程序组件的名称。其实，也可以使用“xyzzzy”而不是使用 `SetTitleString` 来表示在窗口中显示标题消息的行为。计算机并不关心相关行为叫什么名字，只要名字前后保持一致，并且能加以区分就行。然而，我并不是仅仅是为计算机编写程序，而是为初学者编写程序，初学者需要通过了解程序所完成的操作来学习代码，然后才可以开始编写自己的程序。



动手实践

修改消息内容

单击 Solution Explorer 中的箭头，打开 `BeginToCodeWithCSharp` 项目和 `My Snaps apps` 文件夹。然后双击 `MyProgram.cs` 文件，将其打开进行编辑(如果该文件还没有打开)。此时，可以通过更改该程序创建你自己的第一个程序，从而使其以不同的方式工作。可以从更改程序所显示的消息开始。正在运行的程序代码如下所示(从图 1-9 中也可以看到)：

```
public class MyProgram
{
    public void StartProgram()
    {
        SnapsEngine.SetTitleString("Begin to Code With C#");
        SnapsEngine.DisplayString("Welcome to the world of Snaps");
    }
}
```

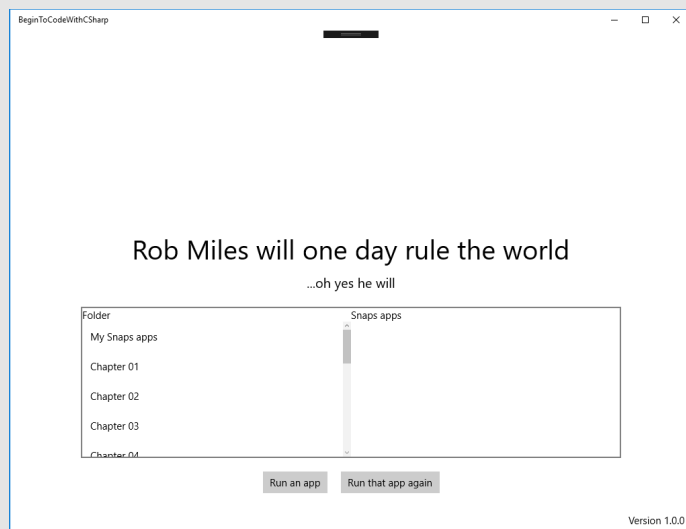
Visual Studio 专门用不同的颜色显示代码的不同部分。通常的惯例是在程序代码中以红色显示程序运行时在屏幕上所显示的文本(这个文本称为字符串)。你并不需要手动将该字符串文本变为红色；只要在代码中正确地设置了字符串，就会自动完成(更多内容稍后详细讨论)。可以按照下面所示的代码对相关字符串进行一些更改，注意，不要修改其他任何代码。然后使用 Visual Studio 中的 Run 按钮再次运行程序。此时屏幕上的消息反映了对代码所做的修改。例如，下面所示的屏幕截图显示了更改后的结果。

```
public class MyProgram
{
    public void StartProgram()
    {
```

```

        SnapsEngine.SetTitleString("Rob Miles will one day rule the world");
        SnapsEngine.DisplayString("...oh yes he will");
    }
}

```



在更改文本时要更小心，不要删除程序中用来表示字符串开始和结束的双引号字符(")。如果删除了，就会发现该文本在 C# 程序中不再具有任何意义，同时当尝试运行程序时会收到错误提示。如果发生了以上情况，也不必担心：Visual Studio 编辑器拥有强大的撤销功能，可以撤销对文件所做的更改。只需要按住 Ctrl 和 Z 键，就可以撤销对文件的连续更改。而如果在编辑器(即显示代码的窗口)中多次按下 Ctrl+Z，那么最终会返回到程序的初始状态。

你已经编写了(或者至少是编辑了)自己的第一个程序。现在，如果有人要求你在屏幕上显示一条消息，我想你应该部分知道该怎么做了。当然，我还会对刚才所修改代码的其他部分进行解释，以便帮助你完全理解！

程序员要点

不存在所谓的“专业”程序

此时所运行的是一个“正确的”程序(在对 MyProgram.cs 文件完成编辑之后)。如果愿意，可以使用 Visual Studio 内置的工具将该程序提交到 Windows Store，以便世界上的任何人都可以下载和使用(虽然坦诚地讲，我不确定有人会认为该程序有用)。学习编程的人往往想知道自己的编程水平什么时候可以达到“专业”开发人员的水平。答案很简单，只要有人愿意付钱给你为他们编写程序，那么你就是一名专业的开发人员。

虽然，付钱完成某事并不会自动地提升你的编程水平，但至少可以为你的努力提供方向。如果想要他人为你的程序付费，那么就需要确保该程序值得付费。在本书中，提供了许多优秀的编程实践示例，当某人看到你的某一程序时说：“我愿意为该程序付费”，你就可以用所学的知识提供高质量和高价值的程序。

1.4 所学到的内容

在第 1 章，我们建立了一个工作场所，安装了编写程序所使用的 Visual Studio 工具，还查看了一些本书所提供的示例应用程序。

你会发现，从本质上讲，Visual Studio 就是一个“程序员的字处理程序”，在该程序中可以创建和测试软件。此外，还会看到，Visual Studio 使用了解决方案和项目来组织资源和程序代码，将这些资源和代码合并在一起可以生成一个现代应用程序。通过更改程序所显示的消息，创建了你的第一个应用程序。

为了巩固对本章的理解，请考虑一下下面关于计算机、程序和编程的相关问题。

程序和应用程序之间的区别是什么？

当人们说起软件时，会发现单词*程序*和*应用程序*会交替使用。当我谈论某一个程序时，通常描述的是一些告诉计算机做什么的代码。我认为应用程序的概念应该更大，需要进行更多的开发。应用程序将程序代码以及诸如图像和声音之类的资产合并在一起，从而为用户提供一个完整的体验。而程序可以非常简单，甚至只包含若干行 C# 代码行。

Visual Studio 中的项目和解决方案之间的区别是什么？

解决方案是最外层的容器。一个解决方案可以包含多个项目，通常被用来创建一个完整的应用程序或者产品。而*项目*可以包含 C# 源文件，通常是一个解决方案的完整子组件。例如，组成 Snaps 框架的所有 C# 程序文件都被打包为一个项目(即解决方案 BeginToCodeWithCSharp 中的 Snaps 项目)，并且可以在任何需要使用 Snaps 资源的解决方案中使用。当你学习本书的过程中开始编写自己的应用程序时，会多次使用到 Snaps 项目。

为什么需要使用 C# 之类的专门语言为计算机编写程序？

针对该问题我最喜欢的答案是两组短句“光阴似箭，果蝇爱香蕉”。人类一般可以很容易理解前半句指的是飞行的物体，而后半句是关于昆虫以及它们所喜欢吃的午餐。但如果想要计算机理解这两个短句的真正含义，所花费的时间将是非常长的。人类使用语言的方式往往充满了模糊和混乱。但幸运的是，在两个用来接收语言的耳朵之间拥有一个强大的“计算机”(即大脑)，我们在幼年就已经开始花费大量的时间对其进行“编程”。相比之下，真正的计算机只是一个非常简单的思维机器，只有给定了严格和快速的规则，才会更好地工作。编程语言包含一组可以让计算机理解的特定指令，以便计算机可以正确地按照指令运行。

Visual Studio 是编写程序的唯一方法吗？

不是。可以使用许多其他的工具来创建软件。其中一些工具被绑定到一种特定的编程语言，而另一些则更加通用。当然，Visual Studio 是最好的工具之一。

如果破坏了程序，应该怎么办？

一些人担心，他们使用某一程序在计算机上完成的事情可能会以某些方式“破坏”该程序。曾经我也这样担心过，但只要确保所完成的任何操作都有返回的方法，就可以克服这种恐惧感。目前，你已经做好了学习 C# 的准备，因为你知道如何在计算机上安装 Visual Studio，并且正在使用来自下载的.zip 文件夹中的演示代码副本，所以即使出现了某些可怕的错误，并最终破坏程序而导致其无法工作，也不必担心，只需要重新对.zip 文件夹进行解压并再次启动程序即可。