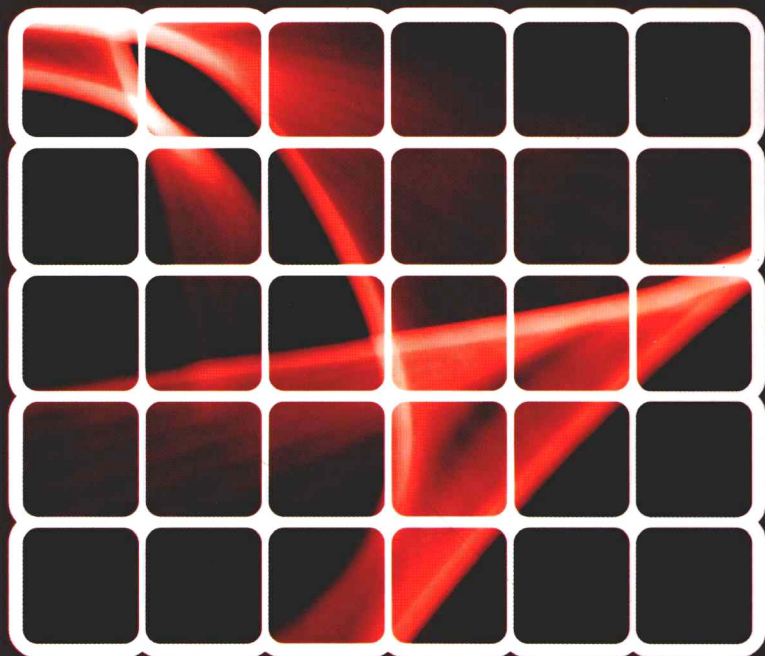




面向应用型高校“十二五”规划教材

# C#程序设计简明教程

陈佳雯 胡声丹 主 编  
陆慰民 主 审



电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

面向应用型高校“十二五”规划教材

# C#程序设计简明教程

陈佳雯 胡声丹 主编  
仲福根 时书剑 副主编  
陆慰民 主审

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

本书遵从突出面向应用型高校学生的宗旨,以应用为主,力图将C#可视化程序设计的方法、面向对象的编程思路、绘图、数据库及文件等高级技术有机结合,帮助学习者不断巩固和检验所学知识,提高程序设计思路和编程能力,使读者通过学习本书,掌握基础知识、开阔视野、培养能力,为后续学习打下坚实的基础。

全书在内容上由教学篇和实验篇两部分构成。其中教学篇共7章,每章后面有若干上机实验,针对本章教学内容供读者检验和巩固所学知识。实验篇内容在章节安排上始终与教学篇保持一致,每章由“知识要点”和“实验题解”两部分构成。

本教材考虑到读者大多为程序设计的初学者,在教材体系上采用由浅入深、循序渐进的方式,适用于应用型高校计算机专业学生的专业课程教学,以及其他各专业学生的公共课程教学。

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

C#程序设计简明教程/陈佳雯,胡声丹主编.—北京:电子工业出版社,2011.10

面向应用型高校“十二五”规划教材

ISBN 978-7-121-14011-2

I. ①C… II. ①陈… ②胡… 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(2011)第132443号

策划编辑:谭海平

责任编辑:侯丽平

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:20.5 字数:524.8千字

印 次:2011年10月第1次印刷

定 价:36.00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](mailto:zltz@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

服务热线:(010)88258888。

# 前 言

随着计算机技术和网络技术的发展, Microsoft 公司于 2002 年推出了 Visual Studio.NET 集成开发环境。从这个版本的 Visual Studio 开始, 微软引入了建立在 .NET 框架上的一种新的编程语言 C# (读做 C Sharp, 意为 C++++) , 它基于 C++ 和 Java 语言, 是用来编写 .NET 框架的语言。

从 20 世纪 90 年代开始, 我国的众多高校针对各专业大学生开设了程序设计公共课程, 其目的是为了培养学生的程序设计思路与编程能力, 教学上以 C/C++ 语言或可视化的 Visual Basic 语言为主。随着近年来 .NET 框架技术的不断进步, 我们发现: C# 语言, 因其兼具 C/C++ 的强大功能及 Visual Basic 的易于使用等特点, 越来越适合取代 C/C++ 和 Visual Basic, 而成为一种新型的适用于教学的编程语言。

本书不仅适用于应用型高校计算机专业学生的专业课程教学, 还适用于其他各专业学生的公共课程教学。考虑到读者大多为程序设计的初学者, 在教材体系上采用由浅入深、循序渐进的方式。遵从突出面向应用型高校学生的宗旨, 以应用为主, 力图帮助学习者不断巩固和检验所学知识, 提高程序设计思路和编程能力。本书力图将 C# 可视化程序设计的方法、面向对象的编程思路、绘图、数据库及文件等高级技术有机结合, 使读者通过学习本书, 掌握基础知识、开阔视野、培养能力, 为后续学习打下坚实的基础。

本书在内容上由教学篇和实验篇两部分构成。其中教学篇共 7 章: C# 编程概述、C# 语言基础、面向对象的程序设计、Windows 窗体应用程序开发、图形图像编程、数据库访问技术和文件操作。每章后面有若干上机实验, 针对本章教学内容供读者检验和巩固所学知识。实验篇内容在章节安排上始终与教学篇保持一致, 每章由“知识要点”和“实验题解”两部分构成; 其中, “知识要点”部分归纳并总结了在教学篇的每章中必须掌握的内容, 而“实验题解”部分则与教学篇中每章后的上机实验保持同步。

关于本书的教学学时, 建议课堂教学 36~54 学时, 实验教学 36~54 学时。教学中应以实验为驱动, 引出语言基础、语法规则、语句结构, 并强调编程思路及能力的培养。为实现课堂讲解与实验结合的教学方法, 培养学生的独立思考及编程能力, 本书实验篇采用 1:1 的形式对教学篇中的上机实验进行解答。其中, 有 50% 的习题在“编程提示”中给出了设计思路, 要求读者独立完成程序的设计与编写工作; 另有 50% 的习题则给出了“实验步骤”, 提供程序源码并做了详尽的分析。

为 C# 语言的初学者考虑, 在所有上课实例和上机实验的分析、解题过程中, 我们尽量采用多种方法, 兼顾界面丰富、知识广泛等特点, 从而启发学生开拓思维, 提高分析问题、解决问题的能力。本教材配有上课实例、实验素材和习题源码等电子课件, 使用本书的学校可与本书作者联系获取相关教学资源, 或者登录华信教育资源网 (<http://www.hxedu.com.cn>) 免费注册后进行下载。

参加本套教材编写的人员有陈佳雯、胡声丹、仲福根和时书剑。同济大学的陆慰民教授审阅了本书, 电子工业出版社的领导和编辑对本书的出版给予了很大的支持和帮助, 在此表

示衷心感谢。

由于作者水平有限，疏漏在所难免，望有关专家和广大读者给予批评指正，在此表示感谢。

本书作者的 E-mail 地址为 [jwnchn@qq.com](mailto:jwnchn@qq.com)，读者若有任何问题或建议，请通过电子邮件联系。

作 者

2011 年 8 月

# 目 录

## 第 1 篇 教学篇

|                              |    |
|------------------------------|----|
| 第 1 章 C#编程概述                 | 2  |
| 1.1 第一个 C#应用程序               | 2  |
| 1.1.1 引例——Hello World!       | 2  |
| 1.1.2 代码分析                   | 4  |
| 1.2 开发环境介绍                   | 6  |
| 1.2.1 启动 Visual Studio 开发环境  | 6  |
| 1.2.2 Windows 窗体应用程序         | 7  |
| 1.2.3 控制台应用程序                | 10 |
| 1.3 窗体和基本控件                  | 12 |
| 1.3.1 控件的基本属性                | 12 |
| 1.3.2 窗体                     | 14 |
| 1.3.3 标签                     | 17 |
| 1.3.4 图像框                    | 18 |
| 1.3.5 文本框                    | 20 |
| 1.3.6 命令按钮                   | 24 |
| 1.4 标识符和关键字                  | 26 |
| 1.4.1 命名约定                   | 26 |
| 1.4.2 关键字                    | 26 |
| 1.4.3 语句                     | 27 |
| 1.5 从程序中输出文本                 | 27 |
| 1.5.1 一个简单的控制台应用程序           | 27 |
| 1.5.2 Write()和 WriteLine()方法 | 28 |
| 1.5.3 格式字符串                  | 29 |
| 1.5.4 多重标记和值                 | 30 |
| 1.5.5 注释                     | 30 |
| 1.6 综合应用                     | 31 |
| 上机实验                         | 33 |
| 第 2 章 C#语言基础                 | 36 |
| 2.1 数据类型和变量                  | 36 |
| 2.1.1 引例                     | 36 |
| 2.1.2 值类型                    | 36 |

|              |                         |           |
|--------------|-------------------------|-----------|
| 2.1.3        | 引用类型                    | 40        |
| 2.1.4        | 变量和常量                   | 40        |
| 2.1.5        | 类型转换                    | 41        |
| 2.2          | 运算符                     | 43        |
| 2.2.1        | 基本运算符                   | 43        |
| 2.2.2        | 条件运算符                   | 45        |
| 2.2.3        | is 运算符                  | 45        |
| 2.2.4        | sizeof 运算符              | 46        |
| 2.2.5        | typeof 运算符              | 46        |
| 2.2.6        | checked 和 unchecked 运算符 | 46        |
| 2.2.7        | new 运算符                 | 46        |
| 2.2.8        | 运算符优先级                  | 46        |
| 2.3          | 分支结构                    | 47        |
| 2.3.1        | if 语句                   | 47        |
| 2.3.2        | if 语句的嵌套                | 49        |
| 2.3.3        | switch 语句               | 50        |
| 2.4          | 循环结构                    | 52        |
| 2.4.1        | for 语句                  | 52        |
| 2.4.2        | while 语句                | 53        |
| 2.4.3        | do/while 语句             | 54        |
| 2.4.4        | foreach 语句              | 55        |
| 2.4.5        | 跳转语句                    | 56        |
| 2.5          | 字符串、数组和集合               | 56        |
| 2.5.1        | 字符串                     | 56        |
| 2.5.2        | 数组                      | 58        |
| 2.5.3        | 集合                      | 62        |
| 2.6          | 运行和调试                   | 68        |
| 2.6.1        | 运行                      | 68        |
| 2.6.2        | 调试                      | 69        |
| 2.7          | 异常处理                    | 71        |
| 2.7.1        | 异常处理类                   | 71        |
| 2.7.2        | try 和 catch 的用法         | 72        |
| 2.7.3        | finally 的用法             | 73        |
| 2.7.4        | 抛出异常                    | 75        |
| 2.8          | 综合应用                    | 76        |
|              | 上机实验                    | 77        |
| <b>第 3 章</b> | <b>面向对象的程序设计</b>        | <b>83</b> |
| 3.1          | 面向对象的程序设计基础             | 83        |
| 3.1.1        | 什么是面向对象的程序设计            | 83        |

|                                   |            |
|-----------------------------------|------------|
| 3.1.2 类和对象的概念 .....               | 83         |
| 3.2 封装和隐藏 .....                   | 84         |
| 3.2.1 定义类 .....                   | 84         |
| 3.2.2 定义类成员 .....                 | 85         |
| 3.2.3 对象及其成员的访问 .....             | 89         |
| 3.2.4 构造函数和析构函数 .....             | 91         |
| 3.3 继承和派生 .....                   | 93         |
| 3.3.1 基类和派生类 .....                | 93         |
| 3.3.2 定义派生类 .....                 | 94         |
| 3.4 重载和重写 .....                   | 96         |
| 3.4.1 重载 .....                    | 96         |
| 3.4.2 重写 .....                    | 99         |
| 3.5 综合应用 .....                    | 100        |
| 上机实验 .....                        | 105        |
| <b>第4章 Windows 窗体应用程序开发 .....</b> | <b>107</b> |
| 4.1 常用控件 .....                    | 107        |
| 4.1.1 单选按钮、复选框和框架 .....           | 107        |
| 4.1.2 列表框和组合框 .....               | 111        |
| 4.1.3 日历和时钟 .....                 | 117        |
| 4.2 菜单和工具栏 .....                  | 123        |
| 4.2.1 引例——记事本程序 .....             | 123        |
| 4.2.2 菜单设计 .....                  | 124        |
| 4.2.3 工具栏设计 .....                 | 128        |
| 4.3 通用对话框 .....                   | 130        |
| 4.3.1 建立通用对话框 .....               | 130        |
| 4.3.2 “打开文件”对话框 .....             | 131        |
| 4.3.3 “保存文件”对话框 .....             | 132        |
| 4.3.4 “字体”对话框 .....               | 132        |
| 4.3.5 “颜色”对话框 .....               | 133        |
| 4.4 多重窗体应用程序开发 .....              | 136        |
| 4.4.1 添加窗体 .....                  | 136        |
| 4.4.2 窗体的实例化和显示 .....             | 137        |
| 4.4.3 窗体的隐藏和关闭 .....              | 137        |
| 4.4.4 不同窗体间的数据访问 .....            | 138        |
| 4.5 综合应用 .....                    | 138        |
| 上机实验 .....                        | 143        |
| <b>第5章 图形图像编程 .....</b>           | <b>147</b> |
| 5.1 GDI+绘图基础 .....                | 147        |



|              |                       |            |
|--------------|-----------------------|------------|
| 5.1.1        | GDI+概述                | 147        |
| 5.1.2        | 坐标系                   | 149        |
| 5.1.3        | Graphics 类            | 150        |
| 5.1.4        | GDI+中常用结构             | 152        |
| 5.2          | 图形绘制                  | 152        |
| 5.2.1        | 线条与形状                 | 152        |
| 5.2.2        | 图形填充                  | 156        |
| 5.2.3        | 文本输出                  | 158        |
| 5.3          | 图像处理                  | 160        |
| 5.3.1        | 加载和显示图片               | 160        |
| 5.3.2        | 裁切和缩放图像               | 160        |
| 5.3.3        | 旋转、反射和扭曲图像            | 161        |
| 5.3.4        | 非规则窗体                 | 163        |
| 5.4          | 综合应用                  | 164        |
|              | 上机实验                  | 169        |
| <b>第 6 章</b> | <b>数据库访问技术</b>        | <b>175</b> |
| 6.1          | 数据库概述                 | 175        |
| 6.1.1        | 关系型数据库模型              | 175        |
| 6.1.2        | 关系数据库标准语言 SQL         | 176        |
| 6.1.3        | 创建数据库与数据表             | 178        |
| 6.2          | ADO.NET 数据访问对象        | 178        |
| 6.2.1        | ADO.NET 简介            | 179        |
| 6.2.2        | 连接数据库: Connection 对象  | 179        |
| 6.2.3        | 执行 SQL 语句: Command 对象 | 181        |
| 6.2.4        | 读取数据: DataReader 对象   | 183        |
| 6.2.5        | 数据适配器: DataAdapter 对象 | 184        |
| 6.2.6        | 数据集: DataSet 对象       | 185        |
| 6.3          | 数据绑定技术                | 189        |
| 6.3.1        | 数据绑定                  | 189        |
| 6.3.2        | 简单数据绑定                | 189        |
| 6.3.3        | 复杂数据绑定                | 190        |
| 6.3.4        | 用代码实现绑定               | 192        |
| 6.4          | 数据库操作                 | 193        |
| 6.4.1        | 数据库操作步骤               | 193        |
| 6.4.2        | 数据库查询                 | 194        |
| 6.4.3        | 数据库编辑                 | 195        |
| 6.4.4        | BLOB 数据处理             | 196        |
| 6.5          | 综合应用                  | 198        |
|              | 上机实验                  | 208        |

|                    |     |
|--------------------|-----|
| <b>第7章 文件操作</b>    | 212 |
| 7.1 文件与文件夹         | 212 |
| 7.1.1 System.IO 类  | 212 |
| 7.1.2 文件夹管理        | 212 |
| 7.1.3 文件管理         | 215 |
| 7.1.4 文件操作控件       | 218 |
| 7.2 用流读写文件         | 220 |
| 7.2.1 流            | 220 |
| 7.2.2 FileStream 类 | 220 |
| 7.2.3 文本文件的读写      | 221 |
| 7.2.4 二进制文件的读写     | 223 |
| 7.3 综合应用           | 224 |
| 上机实验               | 225 |

## 第2篇 实验篇

|                            |     |
|----------------------------|-----|
| <b>实验1 C#编程环境和程序设计初步</b>   | 230 |
| 1.1 知识要点                   | 230 |
| 1.2 实验题解                   | 232 |
| <b>实验2 C#语言基础和语句结构</b>     | 240 |
| 2.1 知识要点                   | 240 |
| 2.2 实验题解                   | 243 |
| <b>实验3 面向对象的程序设计初步</b>     | 259 |
| 3.1 知识要点                   | 259 |
| 3.2 实验题解                   | 261 |
| <b>实验4 用户界面设计和窗体应用程序开发</b> | 267 |
| 4.1 知识要点                   | 267 |
| 4.2 实验题解                   | 270 |
| <b>实验5 图形应用程序开发</b>        | 278 |
| 5.1 知识要点                   | 278 |
| 5.2 实验题解                   | 281 |
| <b>实验6 数据库应用程序开发</b>       | 296 |
| 6.1 知识要点                   | 296 |

|                           |            |
|---------------------------|------------|
| 6.2 实验题解 .....            | 298        |
| <b>实验7 文件的访问与读写 .....</b> | <b>306</b> |
| 7.1 知识要点 .....            | 306        |
| 7.2 实验题解 .....            | 307        |
| <b>参考文献 .....</b>         | <b>316</b> |



第 1 篇

教 学 篇

---



# 第 1 章 C#编程概述

本章通过一个“Hello World!”字幕动画程序，引导读者学习建立 C#应用程序的过程，了解 Visual Studio.NET 集成开发环境的主要特性、编码规范，教会读者创建程序的过程，了解程序文件的组成。通过本章的学习，使读者对 C#编程语言有大致了解，能够编写简单的应用程序。

## 1.1 第一个 C#应用程序

本节从一个简单的“Hello World!”字幕动画程序入手，介绍 C#应用程序的建立过程。

### 1.1.1 引例——Hello World!

**例 1.1** 在装入了背景图像的窗体上，使文字“Hello World!”自左向右移动。移动的方式有两种：

- ① 单击  按钮，文字自左向右移动 5 像素；
- ② 单击  按钮，文字按时钟触发的频率自左向右连续移动，当文字超出窗体右边界时，使其移动到窗体左边界外侧，重新自左向右移动。

程序运行界面如图 1-1 所示。

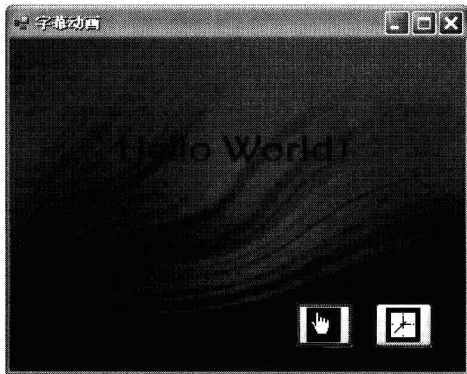


图 1-1 例 1.1 程序运行界面

#### 1. 建立用户界面及属性设置

##### 1) 新建项目

进入 Visual Studio 集成开发环境后，通过“文件”→“新建项目…”命令建立一个 C# 的 Windows 窗体应用程序。

##### 2) 添加控件并设置属性

根据程序要求，展开 Visual Studio 左侧工具箱，在“公共控件”面板中把 Label（标签）、

Button（命令按钮）控件移入窗体，在“组件”面板中把 Timer（时钟）控件移至窗体下方，然后在窗体上设置各控件对象的属性。属性设置参见表 1-1，窗体界面设计如图 1-2 所示。

表 1-1 例 1.1 控件属性设置

| 控件对象名称  | 属 性                   | 设 置 值            |
|---------|-----------------------|------------------|
| Form1   | Text                  | 字幕动画             |
|         | BackGroundImage       | 导入背景图            |
|         | BackGroundImageLayout | Stretch          |
| label1  | Text                  | Hello World!     |
|         | Font                  | 华文新魏，粗体，24pt     |
|         | BackColor             | Transparent      |
| button1 | Text                  | （清空）             |
|         | Size                  | 50 * 40          |
|         | Image                 | 导入“hand.bmp”文件   |
| button2 | Text                  | （清空）             |
|         | Size                  | 50 * 40          |
|         | Image                 | 导入“clock.bmp”文件  |
| timer1  | Enable                | False            |
|         | Interval              | 50 ’ 每隔 0.05 秒触发 |

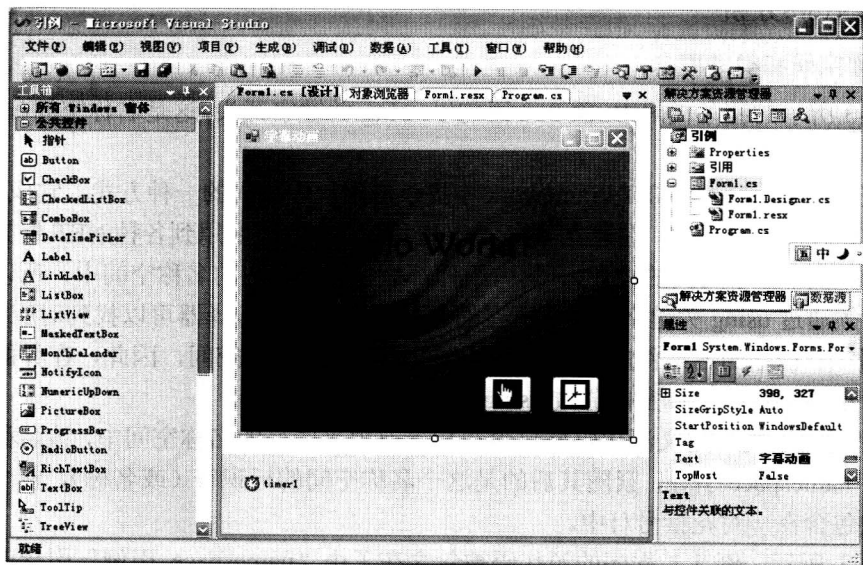


图 1-2 例 1.1 窗体界面设计

## 2. 代码编写

单击“解决方案资源管理器”中的 （查看代码）按钮，切换到代码窗口，在该窗口中编写事件过程，如图 1-3 所示。

## 3. 程序调试

完成设计和代码编写工作后，单击工具栏上的 （启动调试）按钮，进入程序运行状态。单击 或 按钮，进行动画演示。

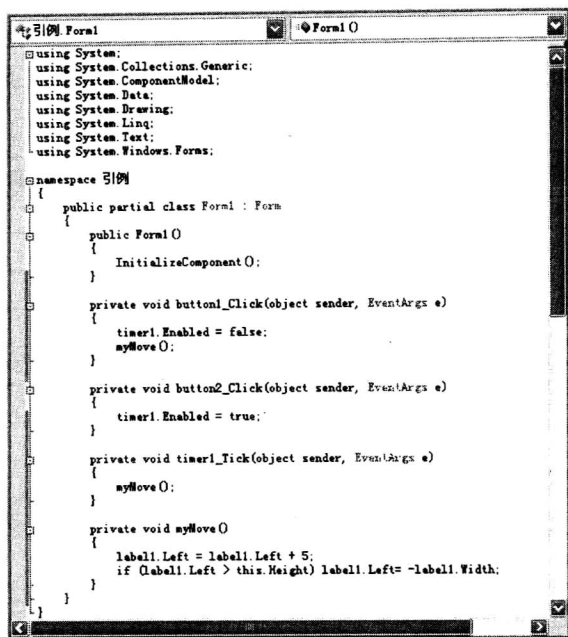


图 1-3 例 1.1 程序代码

## 1.1.2 代码分析

### 1. 名称空间和类的概念

如图 1-3 所示的程序代码中,前 8 个语句以 `using` 关键字开头,这是名称空间(namespace)的引用语句。

名称空间是 Microsoft .NET Framework 中提供应用程序代码的一种方式,它像一个容器,可以唯一标识代码及其内容。在建立 Windows 应用程序时,需要用到各种 .NET 框架类库所提供的组件(如例 1.1 中用到的控件),这些组件都被包含在对应的名称空间中,所以使用这些组件时都必须通过 `using` 关键字引用其所在的名称空间,以使编译器可以找到对应的标识符。

构成例 1.1 程序的各组件的标识符被包含在了不同的名称空间,因此,在界面设计过程中,系统会自动将这些名称空间通过 `using` 关键字引用进来。

在 C# 程序中,用户定义源代码被包含在与当前项目同名的名称空间中,该名称空间的定义由关键字 `namespace` 引导,紧随其后的是这个名称空间的标识符(或名称),整个名称空间中的内容被包含在一对花括号 `{}` 中。

如图 1-3 所示,例 1.1 程序的源代码被包含在了由“`namespace 引例`”引导的一对花括号 `{}` 中,该名称空间的名称为“引例”。在这个名称空间中,通过 `partial class` 关键字定义了一个 `Form1` 类,用户定义的源代码都写在这个类中。C# 是一种面向对象的程序设计语言,所有代码都通过名称空间中的类来实现,在类中定义各类成员,包括类的属性、事件和方法(将在第 3 章中做详细介绍)。

### 2. 初始化窗体

在用户编写代码之前, `Form1` 类中只有一个与类同名的函数 `Form1()`,其源代码如下:

```
public Form1()
{
```

```
InitializeComponent();
```

```
}
```

这个与 Form1 类同名的函数被称为构造函数，构造函数在实例化类对象时被自动调用，对窗体上安排的各控件进行初始化操作（关于类的构造函数的概念将在第3章中做详细介绍）。

### 3. 事件处理

例 1.1 “引例”程序运行后，有两个功能：

- ① 单击  按钮，实现字幕“Hello World!”自左向右移动一次；
- ② 单击  按钮，实现字幕“Hello World!”反复自左向右移动。

如图 1-3 所示，在编写代码时，我们在 Form1 类中输入了以下四段程序源码。其中，名称为 button1\_Click()、button2\_Click()和 timer1\_Tick()的程序段被称为事件处理器（或事件处理程序），而 myMove()的程序段则被称为自定义函数。

```
private void button1_Click(object sender, EventArgs e)
{
    timer1.Enabled = false;
    myMove();
}

private void button2_Click(object sender, EventArgs e)
{
    timer1.Enabled = true;
}

private void timer1_Tick(object sender, EventArgs e)
{
    myMove();
}

private void myMove()
{
    label1.Left = label1.Left + 5;
    if (label1.Left > this.Height) label1.Left = -label1.Width;
}
```

#### 1) myMove()函数分析

先来看一下自定义函数 myMove()，函数中包含两个语句，从上往下依次执行，其作用是使字幕往右移动一次。

其中，语句“label1.Left = label1.Left + 5;”的作用是：将字幕（在标签 label1 中）到窗体左边界的距离增加 5 像素，使字幕右移。而语句“if (label1.Left > this.Height) label1.Left = -label1.Width;”的作用是：判断字幕的位置是否超出了窗体右边界，如超出，则将字幕重新移动到窗体左边界外侧。

#### 2) 事件处理器 button1\_Click()分析

当用户在窗体上单击  按钮（该按钮名称为 button1）时，触发事件处理器 button1\_Click()，依次执行其中的语句。请先跳过第一个语句“timer1.Enabled = false;”不看，当第二个语句“myMove();”执行时，程序调用函数 myMove()从而实现单击  按钮后字幕往右移动一次的功能。



### 3) 事件处理器 timer1\_Tick()分析

如前所述, myMove()函数的作用是使字幕往右移动一次。那么, 如果找到一种方法, 能让用户在单击按钮时, 反复地调用 myMove()函数, 就可以实现字幕的连续右移。

回想界面设计时, 我们曾在窗体中建立过一个时钟对象 timer1。时钟(Timer)有两个重要的属性: Enabled 属性和 Interval 属性。时钟的 Enabled 属性决定了时钟是打开还是关闭的, 在默认情况下是关闭的(即 Enabled = false)。Interval 属性的值是一个整数, 代表了一个时间间隔, 它的单位是毫秒(ms), 其默认值是 100, 代表 0.1 秒。当时钟被打开时(即 Enabled = true), 程序会根据 Interval 属性所指定的时间间隔去反复执行事件处理器 timer1\_Tick()中的语句, 直到时钟再次关闭为止。

因此, 在 timer1\_Tick()事件的源代码中, 写入调用 myMove()函数的语句“myMove();”。只要时钟被打开, 就会反复地执行该句, 使字幕不断右移。

### 4) 事件处理器 button2\_Click()分析

单击按钮(该按钮名称为 button2)时, 触发 button2\_Click()事件, 在其中写入打开时钟的语句“timer1.Enabled = true”, 就能每隔 0.1 秒反复触发 timer1\_Tick()事件, 并反复调用 myMove()函数, 从而实现字幕不断右移的功能。

### 5) 时钟何时关闭?

如果用户单击了按钮, 字幕便不断滚动, 但何时滚动会停止呢? 重新单击按钮时, 这种连续移动的状态被终止。这就需要程序能在用户单击按钮时, 将时钟关闭。所以在 button1\_Click()事件的源代码中, 必须加入“timer1.Enabled = false;”这个语句关闭时钟。如果缺少了这句, 在字幕滚动状态下单击按钮时将无法使之前的连续滚动停止。

## 4. 总结

通过例 1.1 程序的代码分析, 对 C#的 Windows 应用程序运行过程归纳为以下步骤:

- (1) 导入程序所需组件对应的名称空间;
- (2) 创建窗体类对象, 并初始化窗体;
- (3) 运行窗体对象;
- (4) 等待用户行为, 触发对应事件, 执行事件处理程序。

## 1.2 开发环境介绍

Visual Studio 是一个非常复杂且庞大的产品, 但使用它建立应用程序却是非常容易的一件事。本书主要使用 Visual Studio 2008 的版本, 所以首先必须介绍这个开发环境下的一些基础知识。

### 1.2.1 启动 Visual Studio 开发环境

Visual Studio 是一个强大的集成开发环境, 支持 C#、C++、Visual Basic 等多种编程语言。Visual Studio 安装完成后, 初次运行时, 屏幕上会弹出一个“选择默认环境设置”窗口, 如图 1-4 所示。程序员可根据项目需要选择其中一种开发环境, 这个选择会直接影响之后开发的许多方面, 如窗口的布局、控制台窗口的运行方式等。在这里, 选择“Visual C#开发设置”。