第 1 章　C# 语言概述

本章知识目标

- 掌握 Visual Studio 2015 集成开发环境的用法。
- 掌握 Windows 窗体应用程序的创建方法。
- 掌握创建控制台应用程序的方法。
- 掌握简单程序的调试方法。

本章能力目标

- 能够应用 Visual Studio 2015 集成开发环境开发一个简单程序。
- 能够进行简单错误的调试。

Visual Studio 2015 是 Microsoft(微软)公司推出的一套完整的开发工具,用于生成 ASP.NET Web 应用程序、XML Web Services、桌面应用程序和移动应用程序。Visual Studio 2015 包含了 Visual C#、Visual C++ 等几种编程语言,所有的语言使用相同的集成开发环境(IDE),利用此 IDE 可以共享工具并且有助于创建混合语言解决方案。另外,这些语言利用了.NET Framework 的功能,通过此框架可简化 ASP Web 应用程序和 XML Web Services 的开发。

1.1　C# 语言简介

1.1.1　.NET 框架概述

.NET Framework 通常称为.NET 框架,它是一个建立、配置和运行 Web 服务及应用程序的多语言环境,是 Microsoft 的一个新的程序运行平台。.NET Framework 的关键组件为公共语言运行时(CLR)和.NET Framework 类库(包括 ADO.NET、ASP.NET、Windows 窗体和 Windows Presentation Foundation(WPF))。.NET Framework 提供了托管执行环境、简化的开发和部署以及对各种编程语言的集成。

1. 公共语言运行时(CLR)

公共语言运行时是.NET Framework 的基础,是一个在执行时管理代码的代理,它

提供内存管理、线程管理和远程处理等核心服务,并且还强制实施严格的类型安全以及可提高安全性和可靠性的其他形式的代码准确性。代码管理的概念是运行时的基本原则。以运行时为目标的代码称为托管代码,而不以运行时为目标的代码称为非托管代码。

有了公共语言运行时,就可以很容易地设计出能够跨语言交互的组件和应用程序的对象,即用不同语言编写的对象可以互相通信,并且它们的行为可以紧密集成。

2. .NET Framework 类库

.NET Framework 类库是一个与公共语言运行库紧密集成的可重用的类型集合。该框架为开发人员提供了统一的、面向对象的、分层的和可扩展的类库集(API)。可以使用API 开发多种应用程序,这些应用程序包括传统的命令行或图形用户界面(GUI)应用程序,也包括基于 ASP.NET 所提供的最新创新的应用程序(如 Web 窗体和 XML Web Services)。

.NET Framework 可由非托管组件承载,这些组件将公共语言运行库加载到它们的进程中并启动托管代码的执行,从而创建一个可以同时利用托管和非托管功能的软件环境。.NET Framework 不但提供若干个运行库宿主,而且还支持第三方运行库宿主的开发。公共语言运行时和类库与应用程序之间以及与整个系统之间的关系如图 1-1 所示。

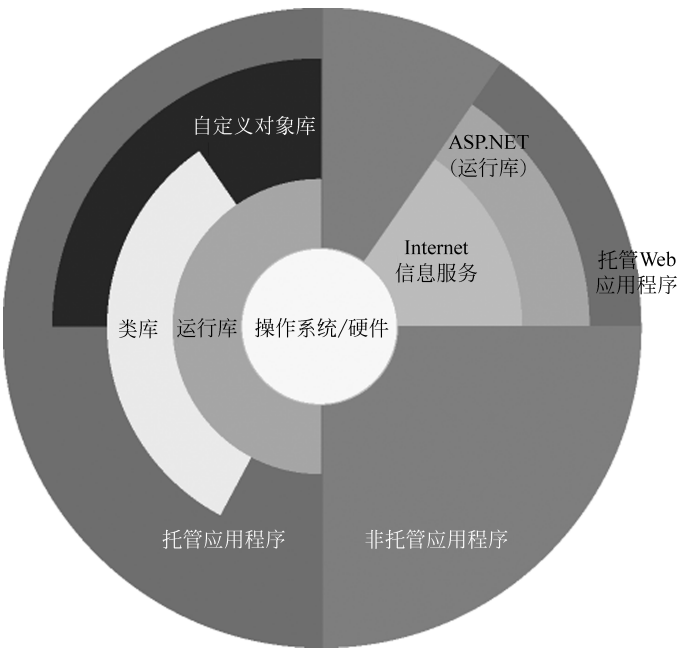

图 1-1 .NET Framework 环境

1.1.2 C#语言的特点

C#(读作 C Sharp)是 Visual Studio 2015 中的一种编程语言,它是由 C 和 C++ 语言

发展而来的，具有更简洁、更先进、类型安全以及面向对象等特点。C#语法表现力强，只有不到 90 个关键字，而且简单易学。C#的大括号语法使任何熟悉 C、C++ 或 Java 的人都可以立即上手。任何一种语言的开发人员通常在很短的时间内就可以开始使用 C#进行高效的工作。C#代码是作为受控代码(managed code)进行编译的，这意味着它们能够得到通用开发语言的服务支持，例如语言互用、冗码剔除、安全性提高以及改进的版本支持等。Visual Studio 支持 Visual C#，这是通过功能齐全的代码编辑器、项目模板、设计器、代码向导、功能强大且易于使用的调试器以及其他工具实现的。通过.NET Framework 类库，可以访问多种操作系统服务和其他精心设计的有用类，应用这些类可显著缩短开发周期。

1.2　Visual Studio 2015 的集成开发环境介绍

Microsoft Visual Studio 2015 的 IDE(集成开发环境)为 Visual C#、Visual C++ 等提供统一的集成开发环境，拥有强大的功能，了解并掌握这些功能可以帮助用户快速有效地建立应用程序。

1.2.1　启动 Visual Studio 2015

启动 Microsoft Visual Studio 2015。在计算机中单击【开始】按钮，选择 Visual Studio 2015，如图 1-2 所示，即可启动 Microsoft Visual Studio 2015。

图 1-2　Microsoft Visual Studio 2015 启动图标

1.2.2　Visual Studio 2015 的集成开发环境

启动 Visual Studio 2015 后，会出现 Visual Studio 2015 集成开发环境，首先是一个【起始页】，如图 1-3 所示。

起始页除了能新建项目和打开项目以外，还包含最近打开过的一部分项目、一些新增功能介绍和供开发者浏览的一些新闻列表。

选择【文件】→【新建】→【项目】→Windows→【Windows 窗体应用程序】命令，选择适当的文件位置及名称后，再单击【确定】按钮，将会出现 Visual Studio 2015 的集成开发环境(IDE)，如图 1-4 所示。集成开发环境由标题栏、菜单栏、工具栏、工具箱、项目设计区、浮动面板区组成。

图 1-3　起始页

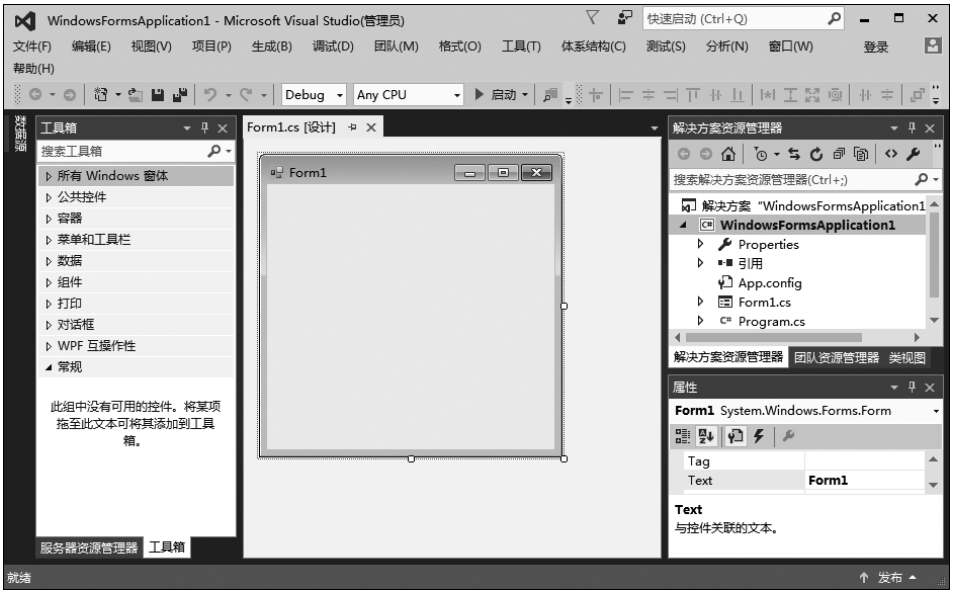

图 1-4　Visual Studio 2015 集成开发环境

1. 标题栏

标题栏位于窗口的最上方,它的作用和其他 Windows 窗口基本一样。标题栏显示项目的名称以及当前程序所处的状态,如“正在运行”等。

2. 菜单栏

菜单栏中的菜单命令几乎包括了所有常用的功能。其中比较常用的【文件】菜单主要用来新建、打开、保存和关闭项目;【编辑】菜单主要用来剪切、复制、粘贴、删除、查找和替换程序代码;【视图】菜单主要是对各种窗口进行显示和隐藏;【调试】菜单主要用来调试程序。

3. 工具栏

工具栏提供了最常用的功能按钮。开发人员熟悉工具栏可以大大节省工作时间,提高工作效率。一般工具栏上面有【标准】工具栏和【布局】工具栏,【标准】工具栏将常用的操作命令以按钮的形式展现,【布局】工具栏是将常用的【格式】菜单命令以按钮形式展现。

4. 工具箱

工具箱(见图 1-5)是 Visual Studio 2015 的重要工具,它提供了进行开发 Windows 应用程序所必需的控件。工具箱是一个浮动的树状控件,它与 Windows 资源管理器的工作方式非常类似,同时展开【工具箱】的多个段(称为“选项卡”),整个目录树在工具箱内部滚动。单击一个名称旁边的加号(+),可以展开工具箱的选项卡;单击一个名称旁边的减号(-),可以折叠一个已展开的选项卡。

图 1-5　工具箱

工具箱显示可以添加到项目中的相应项的图标。每次返回编辑器或设计器时,工具箱都会自动滚动到最近选择过的选项卡和项。当把焦点转移到其他编辑器、设计器或另一个项目时,工具箱中当前选定内容也相应转移。

5. 窗体设计器和【代码】窗口

应用程序设计器为应用程序开发提供一个设计器界面,其中窗体设计器可以设置程序的图形用户界面,而【代码】窗口可以进行代码的编写,如图 1-6 和图 1-7 所示。

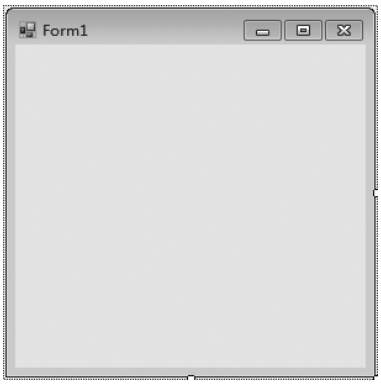

图 1-6　窗体设计器

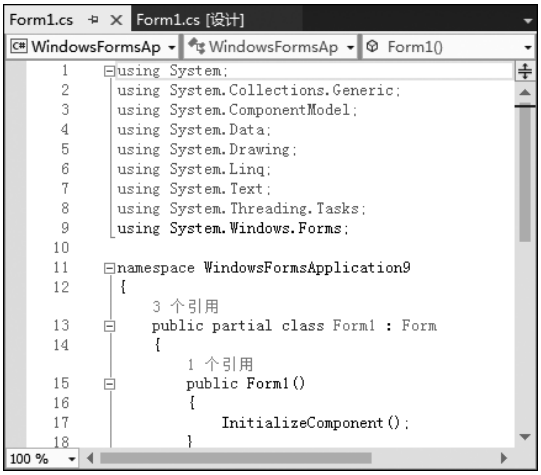

图 1-7　【代码】窗口

6. 解决方案资源管理器

解决方案资源管理器主要用于管理解决方案或项目,利用解决方案资源管理器可以查看项并执行项管理任务,还可以在解决方案或项目上下文的外部处理文件。

解决方案资源管理器利用树形视图(见图 1-8)提供项目及其文件的组织关系,并且为用户提供了对项目和文件相关命令的便捷访问。从该视图中可以直接打开项目项进行修改和执行其他管理任务。由于不同项目存储项的方式不同,解决方案资源管理器中的文件夹结构不一定会反映出所列项的物理存储。与此窗口关联的工具栏提供适用于列表中突出显示的项的常用命令。

7.【属性】面板

【属性】面板用来查看和设置位于编辑器与设计器中选定对象的属性以及事件。可以单击【视图】菜单的【属性】命令来打开该面板,如图 1-9 所示。

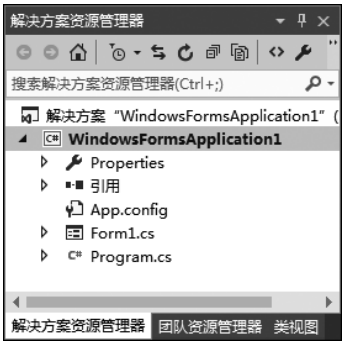

图 1-8　解决方案资源管理器

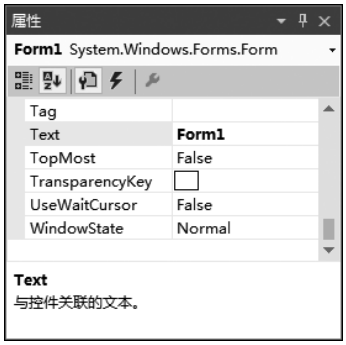

图 1-9　【属性】面板

说明：如果主窗口中没有【解决方案资源管理器】，可以在【视图】菜单中选择【解决方案资源管理器】命令来打开它。同样，工具箱等也可以在【视图】菜单中选择相应命令来打开。

1.3　窗体和基本控件

在 Visual Studio 2015 集成开发环境中我们接触到了窗体和控件，其中窗体是向用户显示信息的可视界面板，而控件则是 Visual Studio.NET 编程的基础，是构成用户界面的基本元素，标签、按钮、文本框等是基本控件。每一个控件在 Visual Studio 中都是一个对象，要使用这些控件，通常要设置控件的属性以及建立事件，有时编程时还要用到控件的方法。

- 属性：是对象所具有的一些可描述的特点，如大小、颜色等。
- 事件：是对象对某些预定义的外部动作进行响应，如单击按钮、移动鼠标等。
- 方法：系统已经提供的一种特殊的子程序，用来完成一定的操作，如文本框光标的定位等。

1.3.1　窗体

窗体(Form)是向用户显示信息的可视图面板，是开发 Windows 桌面应用程序的基础。窗体实质上是一块空白板，开发人员可以通过添加控件来创建用户界面，并通过编写代码来操作数据，从而填充这个空白板。

1. 窗体的常用属性

(1) Name：用于设置窗体的名称，如系统默认标签的名称为 Form1、Form2 等。

给控件命名时必须保持良好的习惯，对控件的命名应该做到见名知义。在命名时，首先书写控件名称的简写，后面是描述控件动作或功能的英文单词。英文单词可以是一个，也可以是多个。每个英文单词的首字母应该大写，如 frmLogin，就是给一个用于登录界面的窗体命名。其他控件的命名可以参考附录 B。

(2) Text：用于设置控件显示的内容，通过改变它的值，可以使控件显示不同的内容。

(3) BackColor：用于设置窗体的背景颜色。

(4) ForColor：用于设置窗体的前景颜色。

(5) Size：获取或者设置窗体的大小。

(6) FormBorderStyle：获取或者设置窗体的边框和标题栏样式。

(7) Icon：指示窗体的图标。

(8) WindowState：确定窗体的初始可视状态，可选值为 Normal(普通)、Minimized(最小化)和 Maximized(最大化)，默认为 Normal。

2. 窗体的常用事件

(1) Click(单击)事件：当单击窗体时，将会触发窗体的 Click 事件。

(2) Load(加载)事件：窗体加载时，将会触发窗体的 Load 事件。窗体的 Load 事件在后面的程序中应用得比较多。

(3) FormClosing(关闭)事件：窗体关闭时，将会触发窗体的 FormClosing 事件。

3. 窗体的常用方法

(1) Show：显示窗体。

(2) Hide：隐藏窗体。

1.3.2 标签

标签(Label)控件是最简单的控件。一般来说，应用程序在窗体中显示静态文本时使用标签控件，在运行状态标签控件中的文本为只读状态，用户不能编辑，因此，标签只是用来显示信息提示而已。

1. 标签控件的常用属性

标签也有 Name、Text、BackColor、ForColor 及其他一些属性，部分常用属性的作用说明如下。

(1) Enabled：用于设置对象是否可以使用。有两个选项：True 和 False。

(2) Visible：用于设置控件的可见性。有两个选项：True 和 False。如果将标签的此属性设置为 False，则程序运行时标签将隐藏起来。

(3) Font：用于设置输出字符的各种特性，包括字体的类型、字体大小等。在【属性】面板中可以通过单击属性值右边的小按钮弹出【字体】对话框来设置字体，也可以展开 Font 属性左边的加号来对字体进行设置。

(4) TextAlign：用于设置控件中显示文本的对齐方式，共有 9 个可选项，分别为 TopLeft、TopCenter、TopRight、MiddleLeft、MiddleCenter、MiddleRight、BottomLeft、BottomCenter、BottomRight。

(5) BorderStyle：用于设置标签的边框形式，有 3 个设置值：None 表示无边框，为系统默认值；FixedSingle 表示边框为单直线型；Fixed 3D 表示边框为凹陷型。

(6) AutoSize：用于设置标签的大小是否根据标签的内容自动调整。True 表示自动调整大小，False 表示不自动调整大小。

2. 标签控件的常用事件

标签可用的事件很多，但一般标签只用于显示提示信息，所以一般应用程序中标签不建立事件，但有时也用到标签的 Click 事件。

1.3.3　文本框

文本框(TextBox)控件用于获取用户输入或者显示的文本，一般用于可编辑文本，也可以使其成为只读控件。文本框可以显示单行，也可以显示多行。

1. 文本框控件的常用属性

同样，文本框也有 Name、Text、Font 等属性，这里不做重复介绍，下面只介绍其他属性。

(1) MaxLength：用于设置文本框中最多可容纳的字符数。当设定为 0 时，表示可以容纳任意多个输入字符，最大值为 32767。若设置为其他数值时，则这一数值就是可以容纳的最多字符数(汉字也作为一个字符处理)。

(2) MultiLine：用于设置文本框中是否允许显示和输入多行文本。当将其设置为 True 时，表示可以显示和输入多行文本。当要显示或者输入的文本超过文本框的右边界时，文本会自动换行，在输入时也可以按 Enter 键强行换行。当将其设置为 False 时，不允许显示和输入多行；当要显示或者输入的文本超过文本框的右边界时，将只显示一部分文本，并且在输入时也不会对 Enter 键做换行的反应。

(3) PasswordChar：用于设置文本框是否用于输入口令类文本。如将其值设置为“*”时，运行程序时用户输入的文本只会显示为一个或者多个“*”，但系统接收的却是用户输入的文本。系统默认为空字符，此时用户输入的可显示文本将直接显示在文本框中。

如果将 MultiLine 属性设置为 True，则设置 PasswordChar 属性不会产生任何视觉效果。如果对 PasswordChar 属性进行了设置，则不管将 MultiLine 属性设置为什么值，均不允许使用键盘在控件中执行复制、剪切和粘贴的操作。

(4) ReadOnly：用于设置文本框为只读，用户无法在文本框中输入数据。

(5) ScrollBars：用于设置文本框是否有滚动条，总共有 4 个可选值。

None：表示不带滚动条。

Horizontal：表示带有水平滚动条。

Vertical：表示带有垂直滚动条。

Both：表示带有水平和垂直滚动条。

(6) WordWrap：用于设置多行文本框在必要时是否自动换行到下一行的开始。如果多行文本框控件可以换行，则该属性设置为 True，该值为默认值；如果当用户输入的内容超过了控件的右边界时，文本框控件自动水平滚动，则为 False。如果此属性设置为 True，则不管 ScrollBars 属性设置为什么值，都不会显示水平滚动条。

2. 文本框控件的常用事件

(1) TextChanged 事件：在控件上更改 Text 属性值时引发该事件。

(2) Enter 事件：当文本框成为窗体的活动控件时引发该事件。

(3) Leave 事件：当文本框不再是窗体的活动控件时引发该事件。

(4) KeyDown、KeyPress 和 KeyUp 事件。

KeyDown 事件在首次按下某个键时引发。

KeyPress 事件在文本框具有焦点并且用户按下并释放某个键后引发。

KeyUp 事件在释放某个键时引发。

3. 文本框控件的常用方法

(1) Focus：可以使文本框获得焦点，如 textBox1.Focus()。

(2) Clear：清除文本框中当前显示的所有文本，如 textBox1.Clear()。

(3) Copy：将文本框中选定的文本复制到剪贴板。

(4) Cut：将文本框中选定的文本复制到剪贴板，同时删除选定的文本。

(5) Paste：用剪贴板中的文本替换文本框中的选定文本。

1.3.4 按钮

用户与应用程序交互的最简便的方法就是使用按钮，按钮是绝大多数应用程序都必不可少的。每当用户单击按钮时，就调用 Click 事件处理程序。

1. 按钮控件的常用属性

按钮除了有 Name、Text、Font 等属性外，还有几个常用属性。

(1) Image：用于给按钮添加一个背景图片，但该图片不覆盖背景色。

(2) ImageAlign：用于设置图片显示在按钮上的位置，分为 TopLeft、TopCenter、TopRight、MiddleLeft、MiddleCenter、MiddleRight、BottomLeft、BottomCenter、BottomRight 9 个属性值，用户可以直接选择。

2. 按钮控件的常用事件

按钮的常用事件是 Click 事件，几乎所有的 Windows 窗体应用程序都使用按钮的 Click 事件。按钮没有 DoubleClick 事件。

1.4 学习任务 1 登录界面的设计

1. 任务分析

本学习任务需要建立一个登录界面，通过输入用户名和密码，单击【登录】按钮，将用户的信息显示出来，具体效果如图 1-10 所示。