

图书基本信息

书名：<<精通C# 3.0与.NET 3.5高级编程>>

13位ISBN编号：9787302195528

10位ISBN编号：7302195528

出版时间：2009-7

出版时间：清华大学

作者：丁士锋//朱毅//杨明羽

页数：782

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>



## 内容概要

C#是微软推出的一种面向对象开发语言，其能让已有经验的开发人员轻松上手，并在很短的时间内就可以使用C#高效地进行工作。

这是读者选择C#的原因。

目前最新的C#版本就是.NET3.5框架上的C#3.0。

本书借助VisualStudio2008开发工具，详细介绍了.NET框架中的4大开发利器：LINQ、WPF、WF和WCF。

LINQ是集成查询语言，它在对象和数据之间建立一种对应关系，可以使用访问内存对象的方式查询数据集合。

WPF是C#开发的图形平台，它改变了传统Windows程序员的开发观念和开发方式。

WF不是一个独立的工作流应用程序，它提供了一些类库用于辅助工作流应用程序的开发，并提供了实现工作流应用程序时所需要实现的一些机制，比如持久化、补偿、跟踪机制等。

WCF是远程通讯技术，其传承了SOA设计的优势。

《精通C# 3.0与.NET 3.5高级编程：LINQ、WCF、WPF、WF》实例具有代表性、编码规范、讲解详细，可作为特定技术开发人员的指导教程，也可以让读者更深入地学习.NET3.5框架的原理和组成。

## 作者简介

丁士锋 毕业于国防科技大学计算机学院。

有多年的大型软件系统开发经验，有近8年的.NET软件项目研发经验，擅长C#语言，对.NET框架及其底层机制有深入的理解。

曾经任职于三星电子、诺基亚等公司，从事软件项目开发。

对企业软件的设计与架构有深入的研究和大量经验，主导过多个大型的企业分布式项目的开发。

朱毅 毕业于上海交通大学，获信息工程、计算机科学双学士学位。

之后又获得了复旦大学软件工程硕士学位。

有6年多的.NET研发经验，涉及分布式系统、B/S体系结构系统、C/S体系结构系统的架构和开发。

曾任职于西门子移动通信有限公司，担任项目经理，致力于企业ERP系统的架构。

现任职于惠普有限公司ISEE项目组进行项目开发。

长期活跃于各大技术社区，曾著有《.NET程序员面试指南》一书。

业余时间喜欢阅读和旅游。

杨明羽 毕业于浙江大学计算机科学与技术专业，高级软件工程师。

多年来一直从事软件开发和项目管理类工作，有近10年的软件开发经验。

擅长C#语言，深入理解.NET框架底层机制，长期追踪.NET框架的最新技术。

曾任职于263在线、阿里巴巴等网络公司。

现任职于上海某大型网络科技公司，担任技术总监一职。

## 书籍目录

第1篇 C#面向对象开发基础第1章 .NET框架和Visual Studio 2008 开发工具 21.1 .NET 3.5框架一览 21.1.1 框架包含的开发语言 21.1.2 C#开发语言的特点 31.1.3 C#的运行机制 31.1.4 C#的类库框架 41.2 Visual Studio 2008 开发界面 41.2.1 Visual Studio 2008界面一览 51.2.2 菜单栏 51.2.3 工具栏 61.2.4 工具箱 71.2.5 属性窗口 71.2.6 解决方案资源管理器 71.2.7 服务器资源管理器 81.3 创建C#应用程序 91.3.1 控制台应用程序 ( Console ) 91.3.2 Windows 应用程序 ( Forms ) 101.3.3 ASP.NET 3.5网站 111.4 小结 12第2章 C#的语法知识 132.1 变量、常量和表达式 132.1.1 常量 132.1.2 变量 132.1.3 变量的类型 142.1.4 类型转换 142.1.5 变量的作用域和生存周期 152.1.6 表达式 152.2 运算符 162.2.1 算术运算符 162.2.2 关系运算符和逻辑运算符 162.2.3 赋值运算符 172.2.4 ? 运算符 172.2.5 运算符优先级 172.3 流程控制语句 182.3.1 if语句 182.3.2 switch语句 182.3.3 for语句 202.3.4 foreach语句 202.3.5 while语句 212.4 方法和函数 212.4.1 方法和函数的定义 212.4.2 函数的参数 222.4.3 函数的返回值 222.4.4 函数的重载 222.4.5 委托 232.5 泛型 232.5.1 什么是泛型 232.5.2 泛型的应用范围 242.6 小结 24第3章 C#的面向对象特性 253.1 面向对象编程概述 253.1.1 什么是类和对象 253.1.2 用C#如何创建类 263.1.3 用C#如何创建对象 273.1.4 什么是面向对象编程 273.2 面向对象的技术 283.2.1 接口 283.2.2 用C#如何创建接口 283.2.3 继承 293.2.4 用C#如何实现继承 293.2.5 多态 303.2.6 用C#如何实现多态 313.3 面向对象开发的简单实例 313.3.1 面向对象功能设计 313.3.2 面向对象的类设计 323.3.3 面向对象的应用 323.4 小结 34第2篇 C#中的LINQ开发第4章 LINQ查询基础 364.1 LINQ基础概念 364.1.1 什么是LINQ 364.1.2 如何使用LINQ 374.1.3 LINQ查询 384.2 LINQ查询表达式 394.2.1 查询表达式 394.2.2 用from子句指定数据源 404.2.3 用select子句指定目标数据 414.2.4 用where子句指定筛选条件 444.2.5 用orderby子句进行排序 464.2.6 用group子句进行分组 484.2.7 用from子句进行复合查询 504.2.8 用join子句进行联接 524.2.9 用join子句进行内部联接 524.2.10 用join子句进行分组联接 534.2.11 用join子句进行左外部联接 544.3 LINQ查询方法 554.3.1 IEnumerable接口 554.3.2 Lambda表达式 574.3.3 用Where()方法进行筛选 584.3.4 用OrderBy()方法进行排序 604.3.5 用Skip()、SkipWhile()跳过元素 624.3.6 用Take()、TakeWhile()提取元素 634.3.7 用Max()等对元素进行数值计算 644.3.8 用Distinct()消除集合中相等的元素 664.3.9 用Concat()连接两个集合 684.3.10 用Union()等进行集合操作 694.4 小结 71第5章 LINQ to ADO.NET——支持LINQ 语言与ADO.NET的交互 725.1 LINQ to ADO.NET概述 725.1.1 LINQ to DataSet概述 725.1.2 LINQ to SQL概述 735.2 LINQ to DataSet——实现复杂数据查询 745.2.1 使用LINQ to DataSet 745.2.2 查询单个数据表 755.2.3 查询多个数据表 785.2.4 用查询创建数据表 815.2.5 修改表中字段数据 835.2.6 使用数据视图DataView 845.3 LINQ to SQL——对象模型 875.3.1 对象模型的原理 875.3.2 生成对象关系设计器 885.3.3 对象关系设计器的构成 905.3.4 深入LINQ to SQL类 915.3.5 通过绑定浏览数据库记录 935.3.6 用LINQ查询LINQ to SQL类 955.3.7 修改数据库记录 965.4 小结 98第6章 LINQ To XML——支持LINQ语言与XML语言的交互 996.1 LINQ to XML概述 996.1.1 什么是LINQ to XML 996.1.2 LINQ to XML与DOM 1016.2 构造XML树 1026.2.1 构造简单的XML元素 1026.2.2 构造具有属性的XML元素 1046.2.3 构造具有子元素的XML元素 1076.2.4 用XElement类构造XML树 1086.3 查询XML树 1096.3.1 查找具有特定属性的元素 1096.3.2 查找具有特定子元素的元素 1116.3.3 对XML元素进行排序 1126.3.4 基于上下文查询元素 1136.4 操作XML树 1156.4.1 加载和保存XML树 1156.4.2 添加元素到XML树 1166.4.3 移除XML树的元素和属性 1186.5 小结 122第3篇 WPF美轮美奂技术第7章 WPF编程入门 1267.1 WPF概述 1267.1.1 理解Windows图形系统 1267.1.2 WPF的功能特点 1277.1.3 WPF架构简介 1277.1.4 WPF的类层次结构 1287.2 WPF开发工具简介 1307.2.1 Visual Studio 2008开发工具 1307.2.2 Microsoft Expression Blend设计工具 1317.2.3 XAMLPad标记文本查看器 1327.2.4 ZAM 3D第三方工具 ( 3D模型 ) 1337.3 第一个WPF应用程序 1337.4 WPF应用程序结构 1357.5 WPF与.NET Framework 3.5平台 1357.5.1 Windows Communication Foundation概述 1367.5.2 Windows CardSpace概述 1377.5.3 Windows Workflow Foundation 概述 1377.6 小结 138第8章 XAML标记语言 1398.1 理解XAML 1398.1.1 XAML是什么 1398.1.2 用C#模拟XAML实现的内容 1418.1.3 XAML的种类 1438.2 XAML基础 1438.2.1 命名空间 1448.2.2 代码后置文件 1468.2.3 使用简单的属性和类型转换器 1498.2.4 属性语法与属性元素语法 1508.2.5 标记扩展特性 1528.2.6 附加属性 1538.2.7 特定的字符和空白 1558.2.8 在XAML中使用事件

1578.2.9 使用其他命名空间中的类型 1588.2.10 加载和编译XAML 1608.3 小结 164第9章 Application全局应用程序类 1669.1 应用程序生命周期 1669.1.1 创建Application对象 1669.1.2 创建一个自定义的Application类 1689.1.3 关闭应用程序 1699.1.4 应用程序事件 1709.2 事件执行周期 1719.2.1 Startup应用程序启动事件 1719.2.2 Activated和Deactivated事件——激活与取消激活 1729.2.3 DispatcherUnhandledException事件——处理应用程序未处理的异常 1739.2.4 SessionEnding事件——注销或关闭系统 1749.2.5 Exit事件——退出应用程序 1759.3 Application类的任务 1779.3.1 处理命令行参数 1779.3.2 访问当前的应用程序 1789.3.3 单实例应用程序 1819.4 小结 185第10章 WPF布局 18710.1 理解WPF布局 18710.1.1 WPF的布局原理 18710.1.2 布局的过程 18810.1.3 布局容器 18810.2 使用StackPanel进行简单地布局 18910.2.1 StackPanel示例 18910.2.2 StackPanel布局属性 19110.3 使用WrapPanel和DockPanel 19310.3.1 使用WrapPanel控件布局 19310.3.2 使用DockPanel控件布局 19410.3.3 简单对话框的实现 19610.4 Grid控件 19810.4.1 创建Grid控件 19810.4.2 调整行列尺寸 19910.4.3 编程创建Grid控件 20110.4.4 合并行和列 20310.4.5 窗体分割 20510.4.6 共享尺寸组 20810.4.7 使用UniformGrid控件 21210.5 基于坐标布局的Canvas 21210.5.1 Canvas控件的使用方法 21310.5.2 Canvas的Z轴 21410.5.3 InkCanvas介绍 21410.6 小结 217第11章 内容控件 21811.1 理解内容控件 21811.1.1 什么是内容控件 21811.1.2 Content属性介绍 22011.1.3 内容的对齐属性 22211.2 内容容器控件 22311.2.1 ScrollViewer滚动条控件 22311.2.2 GroupBox组合框和TabItem标签页控件 22711.2.3 Expander可折叠控件 22911.3 装饰控件 23211.3.1 Border边框控件 23211.3.2 Viewbox自动缩放控件 23311.4 小结 235第12章 WPF依赖属性和事件路由 23612.1 WPF中的树 23612.1.1 逻辑树和视觉树 23612.1.2 编程遍历树结构 23812.2 理解依赖属性 23912.2.1 依赖属性的概念 24012.2.2 定义依赖属性 24212.2.3 依赖属性示例 24412.2.4 共享依赖属性 24812.2.5 注册附加属性 24912.3 理解事件路由 25012.3.1 路由事件的概念 25012.3.2 定义和注册路由事件 25312.3.3 事件路由策略 25412.3.4 RoutedEventArgs类型的参数 25712.3.5 附加事件 25812.4 WPF中的事件 25912.4.1 生命周期事件 25912.4.2 键盘事件 26112.4.3 键盘焦点 26412.4.4 键盘状态 26512.4.5 鼠标事件 26612.4.6 捕捉鼠标 27012.4.7 鼠标拖放编程 27212.4.8 文件拖放示例 27512.5 小结 277第13章 标准控件 27813.1 控件基础 27813.1.1 背景色和前景色画刷 27813.1.2 透明度设置 28013.1.3 更改字体 28113.1.4 更改鼠标光标 28213.2 内容控件 28613.2.1 Label标签控件 28613.2.2 Button控件 28713.2.3 CheckBox和RadioButton控件 29013.2.4 ToolTip提示控件 29113.2.5 使用ToolTipService对象ToolTip 29313.2.6 使用Popup控件 29513.3 文本控件 29813.3.1 TextBox控件 29913.3.2 选择文本 30013.3.3 PasswordBox密码框控件 30113.3.4 TextBlock控件 30313.4 列表控件 30513.4.1 ListBox列表框控件 30513.4.2 获取ListBox选定项 30713.4.3 在ListBox中进行多选 30913.4.4 ComboBox下拉列表框控件 31013.5 范围控件 31313.5.1 ProgressBar进度条控件 31313.5.2 Slider滑动条控件 31413.6 小结 317第14章 窗口与导航 31814.1 Windows类简介 31814.1.1 Window类的基本属性 31914.1.2 窗口的生命周期 32214.1.3 模式与非模式窗口 32414.1.4 窗口的定位和尺寸 32514.1.5 保存和恢复窗口位置 32614.2 对话框窗口 32714.2.1 窗口的宿主 32714.2.2 对话框模型 32814.2.3 MessageBox消息框 33014.2.4 通用对话框 33114.3 非矩形窗口 33314.3.1 圆角窗口 33414.3.2 图形窗口 33514.3.3 调整非矩形窗口尺寸 33714.4 一个简单的导航应用程序示例 33814.5 基于页面的用户界面 34014.5.1 Page类简介 34014.5.2 使用Hyperlink超级链接 34114.5.3 片段导航 34214.5.4 NavigationWindow导航窗口 34314.5.5 Frame页面框架 34414.5.6 导航历史记录 34714.6 NavigationService导航服务 34814.6.1 编程控制导航 34814.6.2 导航生命期事件 35014.6.3 导航记录管理 35314.6.4 使用页函数PageFunction 35714.7 XAML浏览器应用程序 36014.7.1 XBAP的特点 36114.7.2 创建XBAP应用程序 36114.8 小结 362第15章 WPF资源、样式和控件模板 36315.1 程序集资源 36315.1.1 添加资源 36315.1.2 获取程序集资源 36415.2 对象资源 36515.2.1 资源集合 36615.2.2 静态和动态资源 36815.2.3 非共享资源 36815.2.4 编程访问资源 36915.2.5 应用程序资源 36915.2.6 系统资源 37015.2.7 使用资源字典组织资源 37115.3 样式基础 37215.3.1 创建一个样式对象 37315.3.2 在样式中设置属性 37415.3.3 根据指定的类型自动应用样式 37515.3.4 多级样式 37615.3.5 绑定事件处理器 37715.4 样式触发器 37815.4.1 创建简单触发器 37815.4.2 创建事件触发器 38115.4.3 数据触发器 38115.5 控件模板 38315.5.1 理解控件模板 38415.5.2 一个按钮模板示例 38415.5.3 模板的绑定 38615.5.4 模板触发器 38715.6 小结 389第16章 形状、变换和画刷 39016.1 WPF图形 39016.1.1 Shape类 39016.1.2 Rectangle和Ellipse类 39116.1.3 Stretch属性 39216.1.4 Line直线对象 39316.1.5 Polyline多段线对象 39416.1.6 Polygon多边形对象 39516.1.7 线帽和线连

接点 39716.1.8 绘制虚线 39816.2 画刷 39916.2.1 LinearGradientBrush线性渐变画刷 39916.2.2 RadialGradientBrush径向渐变画刷 40116.2.3 ImageBrush图像画刷 40216.2.4 VisualBrush可视化对象画刷 40416.2.5 OpacityMask不透明蒙板 40516.3 WPF变换 40716.3.1 应用变换 40816.3.2 TranslateTransform移动变换 40916.3.3 RotateTransform旋转变换 40916.3.4 ScaleTransform缩放变换 41016.3.5 SkewTransform扭曲变换 41116.3.6 TransformGroup组合变换 41216.4 位图效果 41316.4.1 模糊 41416.4.2 凹凸效果 41516.4.3 浮雕效果 41616.4.4 发光和阴影 41716.5 小结 418第17章 几何图形、图像和可视化层 41917.1 路径和几何图形 41917.1.1 线型、矩形和椭圆几何图形 41917.1.2 使用GeometryGroup组合形状 42017.1.3 使用CombinedGeometry结合形状 42217.1.4 认识PathGeometry对象 42317.1.5 用PathGeometry对象绘制直线 42417.1.6 用PathGeometry对象绘制弧线 42417.1.7 用PathGeometry对象绘制贝塞尔曲线 42517.1.8 使用几何迷你语言 42617.1.9 几何图形的裁切 42717.2 绘图 42817.2.1 绘制形状 42817.2.2 绘制图像 43017.2.3 组合绘制 43017.3 可视化层 43117.3.1 在Visual上绘图 43217.3.2 DrawingVisual宿主容器 43317.3.3 使用命中测试 43517.4 小结 437第18章 WPF数据绑定 43818.1 数据绑定基础 43818.1.1 绑定到元素属性 43818.1.2 使用程序代码创建绑定 43918.1.3 绑定多个属性 44018.1.4 绑定的方向 44118.1.5 绑定更新 44318.1.6 绑定到非元素的对象 44418.2 数据库绑定 44818.2.1 创建数据访问组件 44818.2.2 实现数据实体对象 45018.2.3 显示绑定对象 45018.2.4 更新数据库 45218.3 小结 454第4篇 WF工作流编程第19章 WF编程入门 45619.1 WF基础 45619.1.1 WF简介 45619.1.2 WF的开发环境 45819.1.3 第一个工作流示例程序 45919.1.4 WF架构简介 46319.2 使用WorkflowRuntime和WorkflowInstance类 46419.2.1 理解WorkflowRuntime工作流引擎 46419.2.2 理解WorkflowInstance工作流实例 46719.3 工作流类型和创建模式 46919.3.1 WF工作流的类型 46919.3.2 一个简单的状态机工作流示例 46919.3.3 WF工作流的创建模式 47119.4 小结 472第20章 WF活动 47320.1 理解WF活动 47320.1.1 活动的类层次结构 47320.1.2 活动条件类型 47420.2 使用WF的基本活动 47620.2.1 使用IfElseActivity活动 47720.2.2 使用WhileActivity活动 47820.2.3 使用ParalleActivity活动 47920.2.4 使用ReplicatorActivity活动 48220.2.5 使用ConditionedActivityGroup活动 48520.2.6 使用InvokeWorkflowActivity活动 48820.2.7 使用TerminateActivity活动 49120.2.8 使用SuspendActivity活动 49120.3 本地服务和事件驱动的活动 49320.3.1 理解和实现本地服务 49420.3.2 使用本地服务在宿主和工作流之间通信 49520.3.3 事件驱动的活动简介 49820.3.4 EventDrivenActivity和ListenActivity使用示例 49920.3.5 使用EventHandlingScopeActivity活动 50220.4 小结 507第21章 WF服务 50821.1 WF服务基础 50821.1.1 WF服务的分类 50821.1.2 默认的服务类简介 50921.2 WF持久化服务 51021.2.1 准备数据库 51021.2.2 创建工作流 51121.2.3 使用SqlWorkflowPersistenceService 51321.3 WF跟踪服务 51921.3.1 使用工作流跟踪服务 51921.3.2 跟踪服务配置文件 52321.4 小结 524第22章 基于ASP.NET的工作流批核系统 52522.1 工作流批核系统简介 52522.1.1 系统运行效果 52522.1.2 系统基本结构 52822.2 Workflows工作流项目 52922.2.1 基于事件驱动的活动实现 52922.2.2 UserActivity用户活动实现 53122.2.3 SaveWorkItems自定义活动实现 53722.2.4 ApprovedWorkItemWorkflow工作流实例的实现 53822.2.5 工作流项目帮助类的实现 54122.3 ASP.NET宿主应用程序项目 54222.3.1 初始化工作流运行时引擎 54322.3.2 创建工作流实例 54422.3.3 编辑和审核工作申请表单 54622.4 小结 549第5篇 WCF开发第23章 SOA和WCF基础介绍 55223.1 SOA架构介绍 55223.1.1 软件设计思想发展的简要介绍 55223.1.2 什么是SOA 55323.2 WCF简要介绍 55423.2.1 什么是WCF 55423.2.2 WCF体系框架 55523.2.3 WCF基础概念介绍 55723.3 第一个WCF程序 55923.3.1 HelloWorld服务契约的定义 55923.3.2 HelloWorld的宿主程序 56123.3.3 访问HelloWorld服务的客户端程序 56323.4 小结 565第24章 通道模型和绑定 56624.1 WCF通道模型 56624.1.1 WCF通道模型概述 56624.1.2 消息交换模式和通道形状 56724.1.3 数据报模式 56724.1.4 请求-响应模式 57024.1.5 双工模式 57324.1.6 带会话的数据报模式、请求-响应模式和双工模式 57424.1.7 通道形状的改变 57524.1.8 通道性状和上层服务协议 57624.1.9 通道管理器 57724.1.10 ICommunicationObject接口和状态改变 57724.2 标准绑定介绍 58124.2.1 绑定的基本概念 58124.2.2 标准绑定 58124.2.3 设置绑定的方式 58224.2.4 如何选择绑定 58324.3 本机WCF-WCF交互的绑定和地址 58524.3.1 场景概述 58524.3.2 IPC基本概念 58524.3.3 使用netNamedPipeBinding 58624.3.4 netNamedPipeBinding的地址和配置 58724.3.5 netNamedPipeBinding特点总结 58924.4 跨主机WCF-WCF交互的绑定和地址 58924.4.1 场景概述 59024.4.2 TCP协议概述 59024.4.3 Net.Tcp端口共享 59124.4.4 使用netTcpBinding 59124.4.5 netTcpBinding的地址和配置 59224.4.6 netTcpBinding特点总结 59524.5 与WS-I Basic Web服务进行交互的绑定和地址 59524.5.1 场景概述

59524.5.2 SOAP协议概述 59624.5.3 使用basicHttpBinding 59724.5.4 basicHttpBinding的地址和配置  
59824.5.5 basicHttpBinding特点总结 60024.6 与改进Web服务进行交互的绑定和地址 60024.6.1 场景概述  
60124.6.2 改进Web服务协议概述 60124.6.3 使用wsHttpBinding 60624.6.4 wsHttpBinding的地址和配置  
60724.6.5 wsHttpBinding特点总结 60924.6.6 使用wsDualHttpBinding 61024.6.7 wsDualHttpBinding的地址和  
配置 61124.6.8 wsDualHttpBinding特点总结 61324.6.9 使用ws2007HttpBinding 61324.7 使用脱机模式进行  
消息交互的绑定和地址 61324.7.1 场景概述 61424.7.2 MSMQ协议概述 61424.7.3 使用netMsmqBinding  
61524.7.4 netMsmqBinding的地址和配置 61624.7.5 netMsmqBinding特点总结 61924.7.6 使  
用msmqIntegrationBinding 61924.7.7 msmqIntegrationBinding的地址和配置 62024.7.8  
msmqIntegrationBinding特点总结 62224.8 小结 623第25章 定义服务：契约编程 62425.1 契约定义和分类  
62425.1.1 什么是契约 62425.1.2 契约分类 62425.2 服务契约 62525.2.1 服务契约和WSDL 62525.2.2 服务  
契约的重载问题 62725.2.3 定义请求-响应操作 62925.2.4 定义单程操作 63025.2.5 定义双程操作 63225.2.6  
WCF中事件的实现 64025.3 数据契约 64425.3.1 数据契约和XSD 64425.3.2 使用DataContract特性定义数  
据契约 64625.3.3 数据契约的继承 64925.3.4 已知类型的定义 65025.3.5 数据契约的等效性 65525.3.6 数据  
契约的版本控制 65825.3.7 定义必需的数据成员 66025.3.8 数据默认值的发送 66225.3.9 数组和集合的处  
理 66325.4 消息契约 66725.4.1 消息契约基本概念 66725.4.2 强类型消息 66825.4.3 弱类型消息 67425.5 错  
误处理和错误契约 67825.5.1 SOAP消息的错误处理 67825.5.2 服务端未捕获的异常 67925.5.3 包含详细  
异常信息 68125.5.4 捕捉服务异常 68225.5.5 FaultCode和FaultReason的使用 68425.5.6 使用错误契约  
和FaultException 68525.6 小结 688第26章 WCF中的行为 68926.1 实例管理 68926.1.1 实例管理的设置  
68926.1.2 PerCall实例策略 69026.1.3 PerSession实例策略和会话 69226.1.4 Single实例策略 69526.2 并发管  
理 69826.2.1 并发管理的设置 69826.2.2 Single并发模式 69926.2.3 Multiple并发模式 70126.2.4 Reentrant并  
发模式 70326.3 元数据的发布 70626.3.1 HTTP-GET方式发布元数据 70626.3.2 MEX终节点方式发布元数  
据 70826.4 事务管理 71026.4.1 事务的基本概念和特性 71026.4.2 单服务事务 71026.4.3 分布式事务  
71526.4.4 事务协议和事务管理器 72126.5 小结 723第27章 安全 72427.1 常用概念介绍 72427.1.1 身份验  
证 72427.1.2 授权 72527.1.3 保密性 72527.1.4 完整性 72527.1.5 凭据 72527.2 传输安全性 72627.2.1 传输安  
全性的三要素 72627.2.2 WCF支持的传输安全模式 72627.2.3 传输安全模式的配置 72727.2.4 Transport安  
全模式下的凭证 72827.2.5 Message安全模式下的凭证 72927.3 局域网内Windows平台系统的安全性  
72927.3.1 场景分析 72927.3.2 服务定义 73027.3.3 身份认证 73227.3.4 授权 73427.4 跨Internet系统的安全  
性 73827.4.1 场景概述和绑定选择 73827.4.2 消息安全 73927.4.3 身份认证 74227.4.4 授权 74427.5 安全机  
制的日志和跟踪 74527.6 小结 746第28章 自动点滴管理系统 74728.1 需求分析 74728.1.1 系统介绍  
74728.1.2 客户端功能 74728.1.3 服务端功能 74828.2 系统设计 74828.2.1 服务边界和接口 74828.2.2 传输  
和寄宿的设计 74928.2.3 界面的设计 74928.3 系统实现 75028.3.1 服务契约的定义 75028.3.2 数据契约的  
定义 75128.3.3 服务的实现 75628.3.4 控制台实现 76228.3.5 客户端实现 77028.4 运行和测试 77828.5 小结  
778



## 章节摘录

第1章 .NET框架和Visual Studio 2008开发工具C#（读作C Sharp）语言是.NET框架重点推出的开发语言，其具备C++语言的安全性和Visual Basic（以下简称VB）语言的快速开发特点，是目前最流行的开发语言之一。

因为C#语言的类库全部封装在.NET框架中，所以在讲解具体的C#开发语言前，本章会先介绍.NET框架，其目前的最新版本是3.5。

在介绍完框架后，为了方便程序开发，本章还会简单介绍C#的开发工具Visual Studio 2008简称VS 2008。

1.1 .NET 3.5框架一览.NET 3.5是Microsoft推出的最新开发框架，其支持目前最流行的开发语言VB和C#，也封装了一些常用的类库和组件，通过此框架，可以开发和运行常见的Windows程序和Web程序。本节简要介绍框架的内容和运行机制。

%注意：.NET 3.5框架之前有个.NET 3.0框架，但并没有流行起来。

.NET 3.5包含了.NET 2.0框架和.NET 3.0框架的所有内容。

1.1.1 框架包含的开发语言Java语言也是很流行的一种语言，但其必须运行在虚拟机上，这样可以实现一些跨平台的应用。

而现在最新版本的Visual C# 2008和Visual Basic 2008则必须运行在.NET 3.5框架上。

如果要在服务器上运行Visual C# 2008的程序，则必须在服务器上安装.NET 3.5框架。

%提示：Microsoft最新的操作系统Vista自带.NET 3.0框架。

虽然.NET 3.5是一个成熟的框架，但其底层类库依然调用的是.NET 2.0以前封装好的所有类。

图1.1所示的是.NET 3.5框架的基本组成，最上层就是其支持的开发语言。

在.NET 3.5支持的开发语言中，C#和VB最流行。

VB一般用来快速开发，在小型Windows应用系统中最常用。

C#是Microsoft重点推出的开发语言，其结合了C和C++的一些优点，然后又去除了指针等难于理解的概念，是一门易于上手和开发的语言。

图1.1 .NET 3.5的框架基本组成1.1.2 C#开发语言的特点C#是最流行的开发语言。

相比较其他语言，其具备简单、方便、快速开发等优点，主要特色如下所示。

\* C#语法与C、C++类似，适合刚毕业的学生入门。

\* C#支持面向对象开发，并有.NET底层类库的支持，可以轻松创建对象。

\* C#的高开发效率。

C#的开发工具VS 2005支持拖放式添加控件，开发人员可以轻松完成桌面的布局。

\* C#通过内置的服务，使组件可以转化为XML网络服务。

这样就可以被其他程序调用，也可以被网络上其他机器的其他程序调用，实现了一次创造、重复利用的高效开发模式。

\* XML语言是一种最流行的数据描述语言。

C#提供了对XML的强大支持，可以轻松地创建XML，也可以将XML数据应用到程序中。

\* 自动的资源回收功能，不用再像C++一样，为程序运行中的内存管理伤脑筋。

\* 类型安全是编写代码优先考虑的问题。

C#提供的类型安全机制，可以避免一些常见的类型问题，如类型转换、数组类型越界等。

\* 在.NET框架中，C#可以自由地和其他语言（VB、J#等）进行转换。

1.1.3 C#的运行机制如果只是学习用C#编写一段程序，那很简单，短短几行代码便可以实现；而了解其真正实现的原理，则有助于开发出安全、便于重构的高性能程序。

.NET框架下所有的语言，实际上都是把代码翻译成中间语言（简称MSIL），然后生成标准程序集。无论是VB语言还是C#语言，它们使用的命名空间和类库都是中间语言书写的，所以能够相互操作和相互调用。

从本质上讲，就是在个别语言之下加上了一个共同解释。

.NET中的CLR（公共语言运行时），用来运行生成的MSIL，其实就是将MSIL转换成COM以执行程序

。目前CLR只能在Windows平台上运行。  
综合上面所述，下面就是一个C#程序编译运行的简单步骤。

(1) 将编写的C#程序翻译成中间语言。

(2) 经过C#编译器生成程序集 (\*.exe/\*.dll)。

编译可以使用.NET框架提供的CSC命令，也可以在开发工具Visual Studio中自动编译。

(3) 由公共语言运行库 (CLR) 执行程序集，生成本地代码。

其中CLR的工作流程如图1.2所示。

图1.2 CLR的工作流程1.1.4 C#的类库框架.NET 3.5 提供了开发所有应用需要的类库，但其底层的基础类库依然是.NET 2.0的类库。

类库的主要分类如图1.3所示。

在学习类库的时候，不需要全部掌握其应用技术。

了解类库的基本架构和实现原理才是最重要的。

图1.3 类库的主要分类1.2 Visual Studio 2008 开发界面完善的开发界面可帮助开发人员提高开发效率，这也是VS系列开发工具最大的特点，其完全支持拖动方式设计窗体布局，还可以自动生成各种窗体设计代码。

本例从整体布局方面入手，介绍Visual Studio 2008 IDE开发界面。

1.2.1 Visual Studio 2008界面一览选择“开始”|“所有程序”|“Microsoft Visual Studio 2008”命令，打开VS 2008的开始界面，如图1.4所示。

图中标注出了需要注意的各个区域，其中开始学习文档。

提供一些入门案例和文档。

读者除了看书之外，也可以经常阅读这些文档，有助于了解一些更深入的知识点。

VS开发新闻区域，如果连接了网络，则会显示一些最新的VS开发文档和注意事项。

选择“文件”|“新建”|“项目”命令，打开VS 2008提供的“新建项目”对话框，如图1.5所示。

其中一定要注意“选择框架版本”下拉列表框，这里默认是“.NET Framework 3.5”选项，但其支持.NET 3.0和.NET 2.0框架版本。

如果要开发旧版本的程序，可通过此列表选择。

图1.4 VS 2008的开始界面

图1.5 “新建项目”对话框本例以开发Windows窗体应用程序

为例，进入正式开发界面。

选择“模板”区域中的“Windows窗体应用程序”选项，在项目属性区域设置项目的名称和保存位置

。单击“确定”按钮即可进入VS 2008的开发界面。

这个界面在前面已经提到过，本处不再给出图示。

1.2.2 菜单栏菜单栏位于开发界面的顶端，提供一系列默认的工具和可执行操作，如数据库配置工具和测试工具等。

本节只是简要介绍每个菜单的功能，并不学习具体工具的使用。

希望读者对VS 2008的整体功能有一个概括的了解。

\* “文件”菜单：包括项目打开、保存、导出等。

和普通软件的文件菜单没有多少区别。

\* “编辑”菜单：包括常用的查找、替换、删除等操作。

\* “视图”菜单：视图就是从整体上对开发界面进行布局，包括一些常用的提示窗口。

此菜单非常重要，如果显示一些错误提示窗口和资源管理窗口，则开发人员可以直观地了解程序的错误，以及程序所包括的所有文件。

最常用的视图窗口有“服务器资源管理器”、“解决方案资源管理器”、“错误列表”、“工具箱”和“属性窗口”，这5个窗口尽量在每次打开开发界面时自动打开，以方便浏览整个项目的布局。

\* “项目”菜单：是对整个项目内容的管理。

例如，在项目中添加新的窗体，引用一些其他项目程序，以及查看项目的分布式系统关系图。

\* “生成”菜单：是对整个项目的编译、发布和发布配置。

在项目开发完毕后，可借助此菜单，实现项目的编译和打包。

\* “调试”菜单：是开发人员在编写代码时，用于执行、调试、判断代码，还可以在代码中设置断点，以查看变量的结果。

此菜单是开发人员常用而且必须了解的菜单。

\* “数据”菜单：是对项目中当前的数据源进行管理，这些数据源包括数据库、各种服务和对象等。

如果项目中没有数据源，也可以通过此菜单实现数据源的添加和配置。

\* “格式”菜单：用来调整窗体中各个控件的布局，如对齐、控件间距、大小等。

可一次选择多个控件，进行整体操作。

\* “工具”菜单：提供VS 2008可以支持的所有工具。

如果要用菜单中没有的工具，还可以自行添加。

这些常见的工具包括代码段管理器、宏和服务连接等。

\* “测试”菜单：可能很多读者已经听说过NUnit测试工具，以前它是一个单独的为.NET提供测试的工具，但现在VS 2008集成了这种工具，开发人员可以使用此工具对项目 and 类库进行各种测试，可以及时检查代码错误。

\* “窗口”菜单：提供一些窗口的布局操作，如隐藏、浮动、拆分等，但它不太常用。

\* “帮助”菜单：这里提供了前面安装的MSDN说明文档的一些操作。

%说明：使用VS 2008前，一定要先了解这些菜单内容，知道VS 2008都提供了哪些方便的操作。

1.2.3 工具栏VS 2008提供了多达40多种的工具栏，可实现对数据库、报表、Office文档、常用操作等的各种操作。

为了让读者更方便学习，本例只介绍最简单、最常用的几种工具栏。

选择“视图”|“工具栏”命令可以打开这些工具栏。

\* “标准”工具栏。

和其他软件的标准工具栏一样，提供常见的保存、打开、新建按钮。

其中保存按钮有两个。

按钮用来保存当前打开的单个文档，而按钮用来保存当前项目所有修改后的文档。

\* “布局”工具栏。

用来对窗体中的各个设计组件进行统一布局，如左对齐、居中等。

此工具栏在开发Windows窗体程序时非常重要，而开发Web程序则不需要。

\* “调试”工具栏。

这个工具栏是每个开发人员必须显示的工具栏，用其可以实现对代码的执行、中断、逐行执行等功能。

当鼠标指针指向某按钮时，还会提示这个按钮的快捷键。

开发人员如果熟记这些快捷键，则可以用键盘提高操作速度。

执行代码的快捷键是“F5”键，属于最最常用的代码操作。

\* “文本编辑器”工具栏。

在打开窗体设计视图时，此工具栏处于不可用状态。

因为其只支持代码文本的编辑，包括代码的缩进、注释、标签等。

%提示：针对Windows程序开发和Web程序开发，所使用的工具栏并不相同。

读者可根据自己的项目属性，来决定都显示哪些工具栏。

1.2.4 工具箱工具箱包含了VS 2008提供的常用控件，如按钮、下拉列表框、列表框等。

因为VS 2008提供的控件非常多，所以这些组件被分成了以下常见的几组。

\* 所有Windows窗体：包括创建普通Windows窗体所需要的所有标准组件，如按钮、文本框、状态栏、分割条等。

\* 容器：可以包装其他控件的控件，如Panel、TabControl等。

\* 菜单和工具栏：用来设计窗体布局的一些复杂控件，可实现Windows窗体中的菜单和工具条。

\* 数据：包括数据显示控件和数据源配置控件。

- \* 组件：最复杂的一种控件，包括事件日志管理、进程管理和目录管理等。
- \* 打印：提供多个实现打印功能常见的对话框，如打印对话框、打印预览对话框等。
- \* 对话框：是Windows中常见到的一些对话框，如颜色选择对话框、文件打开和保存对话框等。
- \* 报表：提供了水晶报表的一些控件。

%提示：在代码视图中，工具箱中的组件为不可用状态。

1.2.5 属性窗口属性窗口可用来显示项目、窗体、控件、数据源等所有可视资源的属性。

如果要查看某个按钮的名字和字体等，可通过打开属性窗口来设置。

按下快捷键F4就可以打开属性窗口，其效果如图1.6所示。

在图1.6选中的区域中，有一个按钮，用来设置控件的事件。

如果要查看的资源属性不包括事件，则不会显示此按钮。

如果要显示项目的属性，可先选中项目，然后按下F4键，则自动打开项目的属性，可以在此处设置项目的调试方式和保存位置等。

1.2.6 解决方案资源管理器解决方案资源管理器就类似于Windows操作系统的资源管理器。

可以在此窗口下查看当前项目所包含的所有资源，如文件夹、类文件和数据文件等，如图1.7所示。

图1.6 属性窗口

图1.7 解决方案资源管理器图1.7所示是一个标准的Windows应

用程序文档，其中默认生成一个“Form1.cs”文件。

这是一个窗体文件，包括两部分：设计和源代码。

还有一个类文件“Program.cs”，用来设计当前程序的入口。

“引用”文件夹下包含当前程序引用的其他程序集的内容。

如果要引用网络上下载的一些组件，则可以右击此文件夹，在弹出的快捷菜单中选择“添加引用”命令来添加这些组件。

解决方案资源管理器在系统中被保存为一个完整的文档，默认扩展名为.sln。

该解决方案下可以包含多种项目，既可以包含Windows项目，也可以包含Web项目，还可以在Web项目中引用Windows项目。

%提示：对于一个比较庞大的项目，首先设计好整体解决方案，然后依次添加各个项目。

1.2.7 服务器资源管理器服务器资源管理器以前并不常用，但在VS 2008中，其功能被彻底地挖掘出来。

因为VS 2008提供了LINQ to SQL类，此类必须依靠数据源才可以生成数据库表的映射类，而数据源的管理就在服务器资源管理器中。

选择“视图”|“服务器资源管理器”命令，打开服务器资源管理器，如图1.8所示。

从图1.8中可以看出，VS 2008主要提供两种资源：服务器和数据连接。

服务器用来连接当前可以连接到的所有机器，而数据连接可以连接SQL Server 数据库中的所有数据表。

默认生成的数据连接名称为“机器名+数据库名”。

%提示：可通过右击图1.8中的“数据连接”选项，实现新数据连接的添加。

1.3 创建C#应用程序本章开始进入开发的初始阶段，了解了VS 2008的界面和功能后，通过创建一个简单的程序，来学习如何在VS 2008中开发真正的项目。

因为VS 2008提供开发不同项目的模板，所以本节特别提供了3种常见的应用程序类型：控制台应用程序、Windows应用程序和Web应用程序。

1.3.1 控制台应用程序（Console）控制台应用程序是没有界面的程序，运行效果在DOS窗口中，一般用来执行后台代码。

选择“文件”|“新建”|“项目”命令，打开“新建项目”对话框。

选择模板中的“控制台应用程序”选项，单击“确定”按钮，就创建了一个简单的控制台应用程序，如图1.9所示。

主要包括代码输入区、解决方案资源管理器和项目属性3个部分。

从解决方案资源管理器中可以看出，控制台应用程序只有一个文件Program.cs。

此文件包含一个类Program，并且此类中包含一个默认的方法Main。

此方法是所有应用程序的入口处，一定要注意此方法为静态属性static。  
在此方法内输入如下代码，用来输出一段很简单的字符串“中国欢迎你”。  
注意结束语句用分号。

static void Main(string[] args) { Console.WriteLine("中国欢迎你"); }按下Ctrl+F5组合键运行程序，则会弹出一个DOS窗口，并输出字符串“中国欢迎你”，如图1.10所示。

图1.9 控制台应用程序开发界面 图1.10 控制台应用程序输出%提示：按下Ctrl+F5键是运行程序时不进行调试，而按下F5键则是启动程序进行调试。

如果按下F5键可能看不到这个运行的DOS窗口。

1.3.2 Windows 应用程序 (Forms) Windows 应用程序是常见的C/S程序，也就是服务器上安装主程序，然后在各个客户端机器上安装子程序，调用主程序的内容。

一般会在各个客户端机器上安装开启主程序的服务。

这个一般用于内部网络，忽略网速的影响。

在VS 2008中创建Windows 应用程序的步骤如下所示。

(1) 选择“新建”|“文件”|“项目”命令，打开“新建项目”对话框。

选择“Windows应用程序”选项，单击“确定”按钮。

前面已经介绍过新建项目的界面，本处不再给出 图示。

(2) 打开Form1.cs文件，自动打开的是其设计界面。

按下F7键就可以进入窗体的代码视图。

(3) 在代码视图中，默认生成的代码如下所示。

其中，using用来引用本例需要的一些底层类库，而namespace是本项目所在的命名空间。

本窗体的名字是Form1，其实也是一个类。

```
using System;using System.Windows.Forms;//省略部分引用.....
```

```
...namespace WindowsFormsApplication1{ public partial class Form1 : Form { public Form1() {  
InitializeComponent(); } private void Form1_Load(object sender, EventArgs e) { } }}(4
```

) Form1\_Load是在加载窗体后触发的事件，本例要输出的内容就写在此事件中。

在此处输入如下代码，其中，MessageBox是System.Windows.Forms的一个类，用来弹出一个对话框。

而Show是此类的一个方法。

MessageBox.Show("中国欢迎你"); (5) 按下F5键运行此项目，结果如图1.11所示。

图1.11 Windows应用程序输出%注意：运行程序后弹出的对话框一定会在窗体之前显示，因为在输出语句“MessageBox.Show”时，加载窗体的事件还没有完成。

1.3.3 ASP.NET 3.5网站ASP.NET 3.5网站就是常说的Web程序，也是B/S结构的一种程序，其将主程序布置在服务器上。

而客户端机器只要有浏览器就可以了，不需要安装和设计任何单独的程序。

ASP.NET程序的版本一般随着.NET 框架的版本变化而变化。

例如，在.NET 2.0下，通常称为ASP.NET 2.0应用程序；而在.NET 3.5版本下，则通常称为ASP.NET 3.5应用程序。

本例要创建一个输出“中国欢迎你”的Web页面，实现步骤如下所示。

(1) 选择“文件”|“新建”|“网站”命令，打开“新建网站”对话框，如图1.12 所示。

(2) 在模板区选择“ASP.NET网站”选项，其他使用默认设置。

然后单击“确定”按钮，打开Web程序的开发界面，如图1.13所示。

图1.12 “新建网站”对话框 图1.13 Web程序的开发界面 (3) 默认生成一个Default.aspx文件，这是一个默认页面，其后台代码保存在Default.aspx.cs中。

项目中还生成一个web.config文件，用来配置网站的一些安全和个性化设置，如是否启用角色管理、是否使用Cookie等。

%注意：Default.aspx保存前台代码，而Default.aspx.cs保存后台代码。

(4) 打开Default.aspx.cs后台类文件，其默认代码如下所示。

其中，关键字partial表示此处的类“\_Default”，仅仅是该类的一部分，还有一部分用来表示界面的内

容，并不在此处显示。

```
using System;using System.Web.UI;//省略部分引用.....public partial class _Default :  
System.Web.UI.Page { protected void Page_Load(object sender, EventArgs e) { }} (5) Page_Load是页面加  
载时触发的事件，在此处输入下面的代码，用来在页面中输出字符串。  
Response.Write("中国欢迎你"); (6) 按下F5键运行程序，效果如图1.14所示。
```

可以看到运行效果显示在IE浏览器中。

图1.14 ASP.NET 3.5应用程序输出

1.4 小结本章首先介绍了.NET框架的组成和原理，然后从全局方面了解VS 2008的功能。

介绍了VS 2008的界面、菜单栏、工具栏、窗口和工具箱等，最后学习如何使用VS 2008提供的帮助功能。这有利于初学者快速了解.NET的一些类库，因为这些帮助文档提供了中文的帮助说明和应用案例。本章最后通过3个简单的案例，从不同应用程序的角度学习了如何真正地使用VS 2008进行项目开发。

编辑推荐

超值光盘内容： 随书附赠微软Visual Studio 2008学习版安装光盘 6小时多媒体视频讲解  
资深.NET程序员，全新视角，解读.NET3.5框架的最新技术趋势 深入剖析.NET3.5框架的四大开发技术及.NET3.5框架的底层机制

#### 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>