知识改变命运
学习成就未来

附CD光盘

.NET Framework 3.5 开发技术详解

王毅 编著　杨云 审

人民邮电出版社
POSTS & TELECOM PRESS

.NET Framework 3.5 开发技术详解

王毅 编著　杨云 审

人民邮电出版社

北京

图书在版编目（CIP）数据

.NET Framework 3.5开发技术详解 / 王毅编著. —北京：人民邮电出版社，2009.2
ISBN 978-7-115-19091-8

Ⅰ. N… Ⅱ. 王… Ⅲ. 计算机网络—程序设计 Ⅳ. TP393.09

中国版本图书馆CIP数据核字（2008）第168775号

内 容 简 介

本书全面讲解了.NET Framework 3.5 开发中各方面的技术要点，共分 19 章。本书没有对各个功能的工作原理进行深入的揭示，而是采用“知识进述”＋“代码示例”的方式，让读者可以快速体验并掌握.NET Framework 3.5 的开发方法。

本书适合从事各种.NET 程序开发的人员阅读。通过不同层次的例子，相信可以让读者理解、掌握.NET 程序开发，特别是关于 WPF、WCF 和 WF 这 3 种重要框架的应用所要掌握的知识。

.NET Framework 3.5 开发技术详解

◆ 编　　著　王　毅
　审　　　杨　云
　责任编辑　刘　浩
◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
　邮编　100061　　电子函件　315@ptpress.com.cn
　网址　http://www.ptpress.com.cn
　北京鸿佳印刷厂印刷
◆ 开本：787×1092　1/16
　印张：26.25
　字数：643 千字　　2009 年 2 月第 1 版
　印数：1－3 500 册　　2009 年 2 月北京第 1 次印刷

ISBN 978-7-115-19091-8/TP

定价：55.00 元（附光盘）

读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

当前的软件技术领域有几个比较明显的趋势和潮流——富客户端重新受到重视，企业开始大量构建面向服务的应用，基于工作流的管理系统大量的出现。要想使自己开发出的产品或解决方案更加富有竞争力，那么合理地使用符合现在和未来趋势的技术，则成为一种必然的要求，采用.NET Framework 3.x 技术进行开发就是一种可行性很高的选择。

有许多技术人员在面对.NET Framework 3.0/3.5 大量的新知识、新技术的时候感觉到彷徨，对自己现有的技术知识是否过时会产生疑问，对新技术产生一些抵触的情绪。其实作为技术工作者来说，知识的更新一般都是难以避免的。不论是 Java 程序员也好，Linux 程序员也好，甚至是编写汇编代码的程序员都需要不定期地更新自己的知识。Java 程序员需要更新了解各种层出不穷的框架、应用服务器等知识；Linux 程序员需要了解 Linux 系统近期增加的应用程序接口；汇编程序员则需要了解不断推陈出新的 CPU 指令集，可以说不存在不需要更新的知识。其实，对于技术人员来说，只要能学习到真正代表业界趋势的技术，只要能够花不太多的时间掌握一些新技术，新技术也是很可爱的。编写本书的一个目的就是要让读者认识到.NET 3.5 相关的新知识都是基于现有技术，体会、掌握并应用这些新技术并不困难。

如果说面向对象设计、程序架构是技术人员的内功，那么 WPF/WCF/WF 这样具体的开发类库和模型就是技术人员的招式，技术人员的内功最终要依靠招式来体现。为了让读者能够快速掌握 WPF/WCF/WF 这样杀伤力巨大的招式，本书编写时以读者可以快速理解知识点，在较短的时间内能够开始编写相应的应用程序为导向，用平实的语言对技术点进行归纳和概括，每个技术点结合浅显易懂的实例，从而将复杂的技术问题化解在循序渐进的讲述中。

本书内容

本书细致而全面地讲解 Eclipse Web 开发实践，全书分为 5 篇，共 20 章。

第 1～2 章　.NET Framework 3.0/3.5 在.NET 技术体系中的地位和作用。

第 3 章　对 WPF 进行了概念性的讲述。

第 4～5 章　WPF 的布局和控件模型。

第 6～7 章　数据绑定和样式。

第 8～9 章　WPF 的 2D/3D 绘图功能和多媒体支持。

第 10 章　WPF 的高级应用。

第 11～12 章　WCF 的理论知识。

第 13 章　WCF 核心的消息交换机制。

第 14～15 章　WCF 的并发管理和对事务的支持。

第 16 章　不同环境下如何使用 WCF 的安全特性。
第 17～18 章　WCF 的扩展性和 WCF 的运行环境。
第 19 章　各种 Activity 的使用以及工作流的持久化。

本书特点

- 本书对于.NET Framework 3.5 各个重要的知识点进行了详细分析。
- 对 WPF/WCF/WF 的开发细节进行了由浅入深的讲解，特别是 WPF 和 WCF，基本涵盖了这两大技术的所有重要知识点。
- 全书始终采用知识讲述＋代码示例的方式，非常易于理解和学习。
- 作者根据自身的工作经验和学习研究所得，给出了一些原则性的指导。
- 读者不必具有非常深厚的 C#语言基础，也可以在较短的时间内理解本书中提到的技术术语和代码示例；对于有较多经验的读者来说，则可以作为进行开发工作时手边的参考。

致谢

感谢父母在我写作的过程中对我关怀与支持！
感谢杨云在此过程中给予我的帮助！
感谢参加资料整理的各位朋友，他们是：刘小鹿、谢晓锋、蔡芳、胡建、陈雪郊、刘扬、杜华富、胡浩、成梅花、王磊、李渝平、王宇晟、王春中、冉剑、陈代勇、陈佳佳、陈家云、李广平等。

由于水平有限，书中难免存在不足之处，欢迎广大读者批评指正（电子函件：book_better@sina.com）。

作者
2009 年 1 月

目　录

第 1 章　.NET Framework 3.5 简介

随着微软操作系统 Windows Vista 的发布，微软.NET Framework 也随之发展到了一个新的阶段。Microsoft .NET Framework 3.5（简称为.NET 3.5）是.NET Framework 平台的一个新的里程碑。它可以在 Windows Vista、Windows 2003 和 Windows XP 操作系统上运行。

.NET 3.5 是微软.NET 软件平台的最新发展成果。它基于.NET 2.0 的类库和语法，整合了多种新的功能，例如 Windows Presentation Foundation（Windows 界面基础框架），Windows Communication Foundation（Windows 通信基础构架），WorkFlow foundation（工作流基础构架）和 Windows CardSpace。

本章将对 .NET Framework 3.5 及其组件进行整体描述，同时分析其采用的技术，并给出较为详细的说明，目的是让大家对它有一个清晰的了解。

1.1　.NET Framework 3.5 在.NET 技术体系中的位置

与.NET 2.0 给开发者带来大量新鲜而强大的语法特性不同的是，.NET Framework 3.5 并不提供任何语法上的改动，它完全基于.NET 2.0，并在此基础上提供了 4 个强大的类库，这也是本书将重点介绍的内容，如图 1.1 所示。

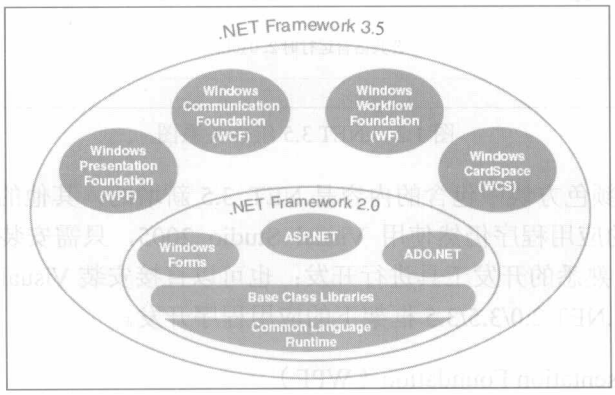

图 1.1　.NET Framework 3.5 的结构

如图 1.1 所示，.NET 3.5 并未对.NET 2.0 现存的技术进行任何改动，包括 ASP.NET、ADO.NET 和 WinForm 在内的主要技术都保持原样，这对熟练掌握.NET 2.0 的技术人员是个好消息，他们所掌握的技术仍然很有价值。根据微软.NET 平台发展趋势来看，.NET 2.0 平台将为.NET 3.5 提供基础类库，.NET 3.5 将着重引入语法，数据访问和 Web 客户端上的一些创新。可以把.NET 3.5 看做是.NET 2.0 的超集，是对.NET 2.0 的一次补充。

如果开发者是从.NET 1.0 或 1.1 迁移到 3.5，那么需要考虑代码的兼容性问题，尽管.NET 框架已经尽力保证向后兼容性，不过由于一些安全方面的改进，仍然有少数重要操作存在不兼容的现象。从.NET 2.0 到 3.5 则完全不存在这个问题，因为.NET Framework 3.5 的所有组件都可在支持 .NET Framework 3.5 的平台上运行。

.NET 3.5 还增加了 LINQ（语言整合查询）和 ASP.NET 的 OR/M 框架（ASP.NET Entity Framework）等新功能。这些新功能增强了 VB 和 C#等语言的作用，为.NET 环境提供了 OR/M 框架，它们都可以和 WPF/WCF 和 WF 共同工作，创建强大的应用程序。

1.2 .NET 3.5 各部分的功能

.NET 3.5 的基本结构在图 1.1 中已经比较清晰地展现出来了。图 1.2 更清晰地显示了.NET 3.5 相关的开发组件之间的层次结构。

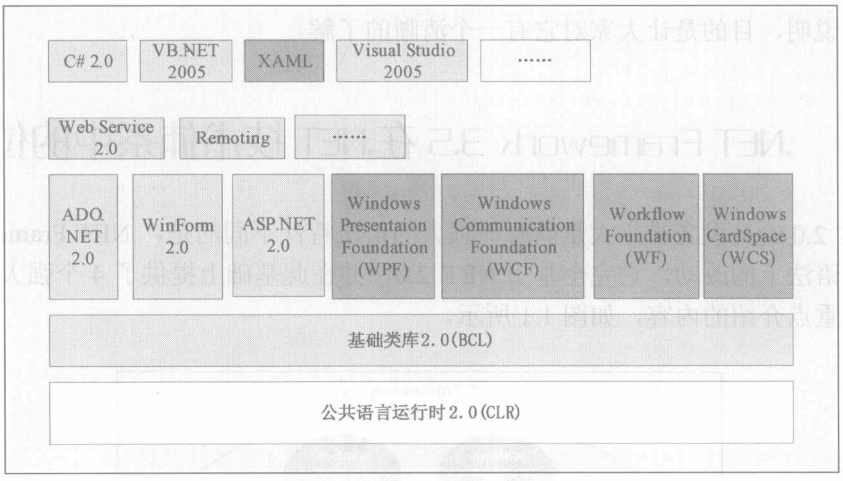

图 1.2 .NET 3.5 组件关系图

图 1.2 中，较深颜色方块中包含的内容是.NET 3.5 新增的，其他的都是继承自.NET 2.0 中。开发.NET 3.5 的应用程序仍然使用 Visual Studio 2005，只需安装相应的开发工具插件和.NET 3.5 即可使用熟悉的开发工具进行开发；也可以直接安装 Visual Studio 2008，Visual Studio 2008 直接支持.NET 2.0/3.5/3.5 框架下的应用程序开发。

1. Windows Presentation Foundation（WPF）

WPF 即是微软潜心开发的新一代用户界面框架。Windows 系统的用户界面的历史可以从 Windows 1.0 开始计算，期间经历了 GDI、GDI+，直到 WinForm，用户界面技术一直都没有得到根本性的改变，只不过是经过了一次次的重构和封装，其本质依然是基于软件模拟渲染，不支持硬件图形加速。与此同时，Windows 游戏的图形界面——DirectX，则得到了飞速的发展，体积阴影、高精度纹理、硬件光源等技术的大规模应用，让 Windows 游戏世界一步一步向乱真的方向发展。由于桌面应用程序和游戏的图形效果差距过大，显卡的图形加速功能被

普通应用程序所浪费，以及用户渐渐对一成不变的界面感到厌倦等原因，微软决心推出新一代的用户界面框架，让普通应用程序的用户界面也能利用 DirectX 中的图形特效，并充分利用图形卡的加速功能，从而带给用户更好的体验。

2. XAML（eXtensible Application Markup Language: 可扩展应用程序标记语言）

XAML 是为 WPF 量身定做的用于用户界面描述的标记语言。XAML 采用了 XML 的格式，它易于阅读，结构规范，也可充分利用 XML 的强大扩展性与其他图形制作工作进行交互。

3. Windows Communication Foundation（WCF）

在.NET 3.5 之前，Windows 操作系统下存在数种分布式消息交换的技术，例如 MSMQ、Web Service、Remoting、企业服务（Enterprise Services）、WSE 等，这些技术所涉及的编程模型也千差万别。微软创建 WCF 的其中一个目的就是统一 Windows 平台的分布式通信技术，让所有这些技术以统一的模型对外提供服务。这种统一的模型让.NET 框架成为了更完善的面向服务（SOA）的平台。

4. Workflow Foundation（WF）

工作流基础类库是新增加的大型基础类库。在此之前，微软从未为外界提供过类似的开发包，因此这也是一次尝试。根据笔者的观察和试用，该开发平台相当强大，足以作为工作流引擎的基础平台。

1.3 .NET 3.5 的组件

1.3.1 Windows Presentaion Foundation（WPF）

WPF 是一个全新的 UI 体系结构，它不仅能比以往的 UI 构架做得更多，还能做得更好、更容易。在 WPF 中，可以发现多种用户界面技术的痕迹。例如 GDI 和 GDI+，这一点其实毫无疑问，毕竟 WPF 是 GDI、GDI+的接班人。又如 HTML，WPF 引入了 XAML 作为界面描述语言，显然受到了 HTML 广泛应用的影响。引入 XAML 进行 UI 描述以后，Windows 程序就可以采用 ASP.NET 那样的代码后置，将界面和程序逻辑分离。WPF 处理动画的方式显然吸取了 Flash 时间线（Timeline）的优点并发展出其独特的 StoryBoard 系统。最后，必须提到的是，WPF 为用户界面的 3D 化提供了强大的支持。WPF 之前，若要在用户界面中提供 3D 元素，必须采用 2D 模拟 3D 的办法，或者采用 DirectX/OpenGL 渲染的方式。这两种方式存在的问题是：2D 模拟 3D 的性能非常低，不可能为 3D 元素提供更多特效；采用 DirectX 或 OpenGL 模式实现的用户界面不容易与其他 Windows 界面元素进行交互。而在 WPF 中，这些都不再是问题，因为 WPF 采用 DirectX 9.0C 渲染 3D 元素，不仅原生地支持了 3D 元素，而且由于采用硬件加速，大大提升了用户界面的显示效率，使得开发人员有机会为用户界面提供更多、更酷的效果。

WPF 吸取了多种技术的优点创造出一个新型的平台，而且在应用程序类型的支持种类上也比其前辈们有不少提高。除了传统的 Windows 窗口程序以外，WPF 还支持浏览器应用程

序（XBAP）。XBAP 可以在客户端的浏览器中运行，它受到浏览器的安全性限制。WPF 还吸收了 Web 程序的特点——功能按页面进行分离，新增了 Navigation Application（浏览型应用程序），可以让用户在同一个窗体中，如同访问 Web 页面那样执行程序，这个特性将在后面章节中进行较为详细的讲解。

注意：WPF 还有一个子集，叫做 Silverlight（开发代号为 WPF/Everywhere），它将作为一个浏览器的客户端技术运行在多个平台的浏览器中。在本书开始编写时，Silverlight 1.0 版本的 beta1 才被公布不久，因此本书不对它进行更多说明。感兴趣的读者可以自行查阅微软站点的资料。

虽然 WPF 是一个全新的 UI 框架，但是为了在最大程度上使 WinForm 的知识得以保留，开发人员仍然可以用相似的代码结构来创建窗体和控件。下面的两段代码分别使用 Winform 和 WPF 技术创建一个窗体和一个按钮。Winform 代码如下：

```
using System.Windows.Forms;
using System;

class program
{
[STAThreadAttribute()]
public static void Main()
{
    Form form = new Form();
    form.Text = "Winform";
    Button btn = new Button();
    btn.Text = "click me";
    btn.Location = new System.Drawing.Point(100, 70);
    form.Controls.Add(btn);
    Application.Run(form);
  }
}
```

WPF 代码如下所示：

```
class program : Application
{
      [STAThreadAttribute()]
      public static void Main()
      {
          program app = new program();
          Window window = new Window();
          window.Title = "WPF Window";
          Button btn = new Button();
          btn.Width = 100; btn.Height = 40;
          btn.Content = "Click me";
          window.Content = btn;
```

```
            app.Run(window);
        }
    }
```

两段代码的结构大同小异，分别定义了一个 Form 和 Window 对象作为主窗口，然后在主窗口中添加了一个按钮对象。稍微有所不同的地方是：在 WPF 中，如果不定义按钮的长和宽，那么默认情况下，按钮会充满整个包含它的容器。读者可以尝试去掉 btn.Width = 100; btn.Height = 40；这两个语句再执行，看看窗口的效果。WinForm 和 WPF 在表面上看起来非常相似，但它们却有着本质上的不同。WinForm 技术实际上只是 GDI 的托管封装，内部仍然使用相同的控件和消息模型。

除了使用 C#和 VB.NET 等语言进行编程以外，还可使用上文提到的 XAML 对窗口进行描述。下面的代码采用 XAML 标记语言编写，它用来描述一个窗口，其功能于上面的代码相同。

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="WPFStart" Height="300" Width="300">
  <Button Content="Click Me" Height="40" Width="100"/>
</Window>
```

如上所示的代码，总体感觉和 HTML 比较类似。Window 标签用于声明一个 Window 对象，xmlns 用于声明一个 XML 的命名空间，xmlns:x 的作用相同，关于它们的详细解释请参见第 4 章。<Button/>标签用于声明一个按钮对象，并且通过按钮对象的 Content 属性指定按钮上显示的内容，可以指定除文字以外的内容，如图片、视频、其他控件容器，甚至可以指定一个按钮，基本上能用在 UI 上的元素都可以指定。

关于 WPF 更多的内容，将从第 3 章开始系统地介绍给读者。

1.3.2　Windows Communication Foundation（WCF）

WCF 除了建立一个统一的分布式通信编程模型以外，还有以下 3 个重要的目标。

（1）基于面向服务的构架，成为面向服务的基础平台。

（2）以更健壮的方式实现 Web Service 和.NET Remoting。

（3）全面实现 Web Service 的标准协议簇——WS-*。

微软创建 WCF 的目的之一是为.NET 平台提供一种统一的分布式计算平台，所以它应当具有广泛的互操作性以及对面向服务构架的直接支持。

微软及许多领先的 IT 企业已经为面向服务（SOA）的平台作好了准备，WCF 即微软公司 SOA 战略中重要的一环。

面向服务的应用程序和先前的分布式应用程序有显著的不同。面向服务应用程序的服务端和客户端是松散耦合的，客户端和服务端不必事先知道对方的存在，只要通过一个 URL，按照约定的通信方式就可以进行互操作。

读者或许会有一些疑问，在.NET 环境中，使用 Visual Studio、NET Framework 及 WSE 可以非常容易地编写 Web Service，创建服务端或客户端，以及和运行在其他平台上的客户端或服务端进行交互。为什么还需要 WCF？

如开发者所知，Web Service 只是.NET 平台上其中一种可以用来构建分布式应用程序的

技术，其他提到过的技术还包括 Remoting、Enterprise 和 MSMQ 等。运用这些技术创建分布式服务的方式和互操作性有很大的区别，而且当试图将分布式计算的基础转换到另一种技术时，代价将会相当大。WCF 就是要解决这个问题。它统一了.NET 环境下分布式计算的编程模式，让开发人员可以不用太关心基础的分布式计算的机制。WCF 不仅可以编写 Web 环境下可以访问的服务，在其他协议的网络中部署服务也很容易。

1.3.3 Workflow Foundation（WF）

Workflow 即工作流，对大多数人来说都是比较新鲜的概念。但实际中，非常多的软件都涉及工作流的处理。在 Workflow Foundation 中，工作流就是用一系列抽象出来的活动（Activities）来描述现实世界中的一个流程。在现实中，工作流往往代表着一件工作从一个人传递到另一个人，并且伴随着状态的改变。一个典型的工作流往往定义了几个必要的步骤，步骤可以是有顺序的，也可以是无顺序的。举例来说，一个员工报销医疗保险费用，其流程就必须是：填写理赔单→提交表单→公司审核→医保机构审核→发放理赔金。这个流程的每个步骤都是必需且有序的，到最终发放理赔金之前，任何一个步骤的失败都将导致流程中断。这个工作流比较简单，路径非常单一，而复杂的工作流中会涉及更多的分支。大多数程序中，涉及的过程处理或多或少都可以看作是广义上的工作流。

工作流由活动元素组成。可以将工作流拆解成为一个一个的活动（Activity）对象，然后按照逻辑组织到一起来描述工作流。

目前，大多数软件对工作流的处理方式都是私有和不开放的，缺乏共通性。不同软件定义的工作流无法被重用和扩展，这对软件的快速更新显然是极为不利的。WF 就是将创建一个工作流所需要的各个环节抽象出来，供开发者使用。WF 并不关心开发者使用 WF 搭建的工作流处理什么具体事务，可以说，只要处理的事务带有流程的性质，就可以应用 WF。

WF 中，工作流可以分为两种：一种是顺序流，另一种是基于状态机的工作流。通常情况下，如果一个流程重点关注的是顺序，而较少涉及对象状态的变化，那么多用顺序流；如果流程非常依赖于对象的状态，而对象的状态在整个流程中会发生多次变化，那么基于状态机的工作流将是更好的选择。

1.4 搭建.NET 3.5 的开发环境

.NET 3.5 附带在 Visual Studio 2008 中，使用 Visual Studio2008 可以很自然地开发.NET 3.5 应用程序。下面分别讲解如何在 windows XP，Windows 2003 以及 Vista 系统上搭建开发环境。

1.4.1 在 Windows XP/2003/Vista 上搭建开发环境

由于在 Windows XP 和 Windows 2003 上搭配开发环境的步骤基本相同，因此在此一道进行讲述。

搭建开发环境的前提条件：

（1）操作系统必须是 Windows XP 家庭版或者专业版，并且是 SP 2 之后的版本（包括 SP2），或者是 Windows 2003 任一版本的 SP1 之后的版本（包括 SP1）。