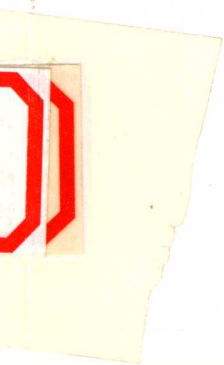
Microsoft®

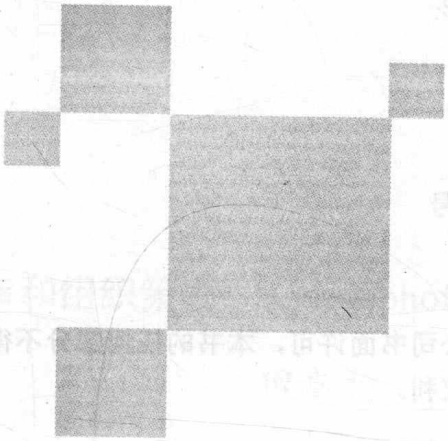
Windows Presentation Foundation(WPF)应用开发

微软公司 著



人民邮电出版社
POSTS & TELECOM PRESS

CD-ROM



Windows Presentation Foundation(WPF)应用开发

微软公司　著

人民邮电出版社
北京

图书在版编目（CIP）数据

Windows Presentation Foundation(WPF)应用开发 / 微软公司著. -- 北京 : 人民邮电出版社, 2011.2
ISBN 978-7-115-23343-1

Ⅰ. ①W… Ⅱ. ①微… Ⅲ. ①窗口软件, Windows－程序设计 Ⅳ. ①TP316.7

中国版本图书馆CIP数据核字(2010)第172994号

版 权 声 明

内 容 提 要

Windows Presentation Foundation（WPF）为开发人员提供了统一的编程模型，可用于构建合并了 UI、媒体和文档的丰富 Windows 智能客户端用户体验。本书介绍如何生成和配置 Windows Presentation Foundation（WPF）的解决方案。

全书共 9 章，讲述了如何生成 WPF 应用程序，在 WPF 中创建 UI，自定义 WPF 应用程序的外观，绑定 UI 控件到数据源，使用集合视图和数据模板显示数据，实现用户控件和自定义控件，创建、打包和打印文档，使用 2D 和 3D 以及多媒体，最后还讲解了如何配置和部署 WPF 应用程序。

本书适合具备.NET Framework 应用程序开发技能，想要使用 Windows Presentation Foundation 在应用程序中实现展示层应用的开发人员阅读。

Windows Presentation Foundation（WPF）应用开发

◆ 著　　微软公司
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
北京艺辉印刷有限公司印刷
◆ 开本：787×1092　1/16
印张：16　　2011 年 2 月第 1 版
字数：382 千字　　2014 年 7 月北京第 5 次印刷
ISBN 978-7-115-23343-1
定价：52.00 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

编审和组织策划： Kyle Uphoff　Foong Chee Ngiam　王　林

田本和　蒋　斌　梁　健

技术编审： 蒋　斌　张　宪　孙　欣　宫　丽

前　言

Windows Presentation Foundation（WPF）是下一代显示系统，用于生成能带给用户震撼视觉体验的 Windows 客户端应用程序。使用 WPF，您可以创建广泛的独立应用程序以及浏览器承载的应用程序。

通过学习本课程，读者能够编写使用 Windows Presentation Foundation 开发丰富的展示层应用。具体技能包括：

- 创建 WPF 应用程序；
- 在 WPF 应用程序中生成 UI；
- 自定义 WPF 应用程序的外观；
- 绑定 UI 控件到数据源；
- 绑定 UI 控件到集合；
- 在 WPF 应用程序中创建新控件；
- 在 WPF 应用程序中管理文档；
- 为 WPF 应用程序添加图形和多媒体支持；
- 配置和部署 WPF 应用程序。

本书结构

本书共分为 9 章。

第 1 章　“使用 Windows Presentation Foundation 创建应用程序”，讲述了如何生成 WPF 应用程序。

第 2 章　“生成用户界面”，讲述了如何在 WPF 中创建 UI。

第 3 章　“自定义外观”，讲述了如何使用样式、模板、触发器和动画自定义 WPF 应用程序的外观。

第 4 章　“数据绑定”，讲述了如何绑定 UI 控件到数据源。

第 5 章　“数据绑定到集合”，讲述了如何使用集合视图和数据模板呈现、排列以及分组数据。

第 6 章　“创建新控件”，讲述了如何实现用户控件和自定义控件。

第 7 章　“管理文档”，讲述了如何创建、打包和打印文档。

第 8 章　“图形和多媒体”，讲述了如何在 WPF 应用程序中使用 2D 图形、3D 图形以及多媒体。

第 9 章　“在 Windows Presentation Foundation 中配置和部署应用程序”，讲述了如何配置和部署 WPF 应用程序。

教学参考资料

获得微软授权使用该教材进行教学的教师，除了可以获得上述材料外，还可以从微软公

司获得如下教学资料。

教参	章节教参与教学大纲
实验与习题答案	实验的具体操作步骤和习题答案
虚拟机	配置好的虚拟机实验环境，用于进行实验和课堂演示
实验室安装指南	包括在实验室部署虚拟机的指南
PowerPoint	用于进行课堂演示

虚拟机

本书案例所提供虚拟机实验均采用微软的Lab Launcher进行操作，Microsoft Lab Launcher是一种虚拟机操作界面，它让用户用一种更简易的操作界面进行虚拟机操作。该工具基于Virtual Server R2 SP1 实现。虚拟机安装完毕后，用户即可使用此界面进行操作，操作界面参见图 0-1。

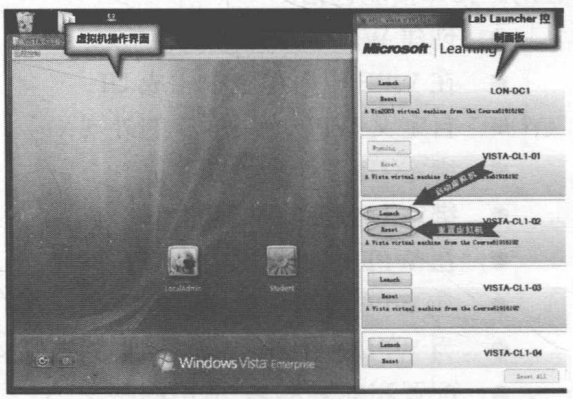


图 0-1　虚拟机操作界面

Lab Launcher 的运行环境要求如下。

- 操作系统为 Windows XP 简体中文版或 Windows Vista 简体中文版。
- 系统中安装.NET Framework 2.0 简体中文版。
- 系统中安装 Microsoft Virtual Server 2005 R2 简体中文版。
- 2GB 物理内存。

在本课程中，共包含一套虚拟机，分别对应不同的内容。

虚拟机	角色
10083A-LON-DEV-01	第 1 章的虚拟机
10083A-LON-DEV-02	第 2 章的虚拟机
10083A-LON-DEV-03	第 3 章的虚拟机
10083A-LON-DEV-04	第 4 章的虚拟机
10083A-LON-DEV-05	第 5 章的虚拟机
10083A-LON-DEV-06	第 6 章的虚拟机
10083A-LON-DEV-07	第 7 章的虚拟机
10083A-LON-DEV-08	第 8 章的虚拟机
10083A-LON-DEV-09	第 9 章的虚拟机

目　录

第 1 章　使用 Windows Presentation Foundation（WPF）创建应用程序

在过去的 10 年中，应用程序的开发模型产生了两个分支，那就是传统的“智能客户端”安装应用程序和基于 Web 的应用程序，两者都为了能在开发界中占据统治地位而展开竞争。这两个模型都要求开发人员牺牲在另一个模型中被视为理所当然的功能。

Web 应用程序提供了更大的作用范围、更简单的部署等功能，但其代价是状态丢失及开发模型更复杂等。同样 Web 应用程序也不能充分利用现有计算机硬件平台丰富的图形图像和多媒体效果。

另一方面，Windows 应用程序确保了应用程序可以脱机工作，并且可以充分利用客户端硬件。但是，也失去了易于部署的特点。同样，Web 开发领域所拥有的大量设计软件和可视化开发工具，也无法用于 Windows 应用程序的界面开发。但显而易见的是，客户端应用程序模型可以为独立或基于浏览器的应用程序提供更好的硬件支持，以实现更复杂的功能。

自从在 Windows 的第一个版本中首次出现以来，构成 Windows 用户体验核心的图形子系统（User32 和 GDI 库）实际上已经诞生了将近 20 年。当然，随着时间的推移，User32 和 GDI 库经历了重大的演变和不断的发展，并且在所有领域都引入了很多新的服务和功能。同时，还出现了诸如 Direct3D 这样的新呈现技术，这些技术能够充分利用在最新的视频卡中公开的图形功能。但是，目前的大多数主流应用程序都没有提供这些图形卡实现的体验。

Windows Presentation Foundation（以前称为“Avalon”）是 Windows 中新的关键图形子系统，它为用户界面、2D 和 3D 图形、文档和媒体提供了统一的方法。它在.NET Framework 基础上生成，并利用 Direct3D 进行基于向量的呈现，为应用程序提供了强大的解决方案。另外，Windows Presentation Foundation（以下简称 WPF）引入 XAML（可扩展应用程序标记语言）。这是一种基于 XML 的语言，用于实例化和填充嵌套对象层次结构。这使设计人员和开发人员可使用诸如 Expression“Sparkle”这样的工具以及第三方专家工具（包括 ZAM 3D 和 Mobiform Aurora），相互协作来完成客户端应用程序的设计和开发。在开发过程中，设计人员可以专注于界面设计而无需关心程序逻辑。

本章讲述了 WPF 的定义、作用和应用基础。

1.1　WPF 概述

WPF 是基于 Windows 操作系统用于创建客户端应用程序的新一代演示系统，可以用它创建从简单的文字处理器或媒体播放器，到企业级业务应用程序等一系列基于 Windows 的应用程序。图 1-1 所示为使用了 WPF 技术构建的范例应用程序界面。

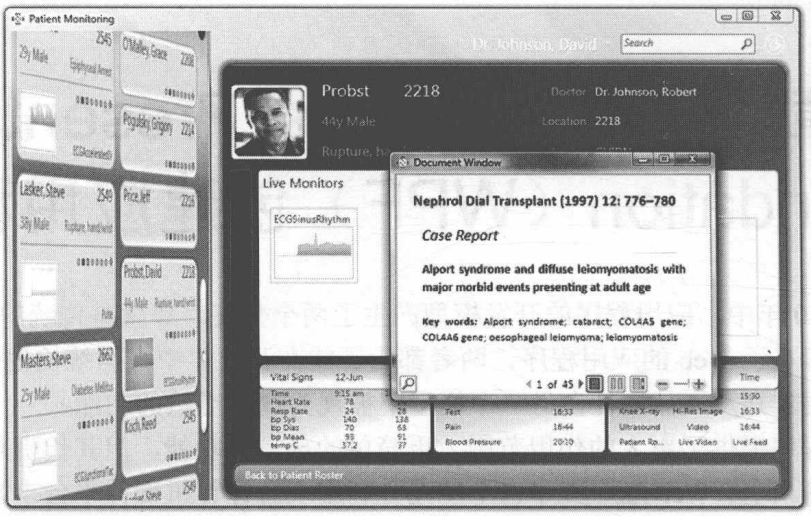


图 1-1　WPF 范例程序

1.1.1　什么是 WPF

WPF 是生成基于 Windows 的应用程序的新基础（Foundation），可以使用媒体、文档和应用程序用户界面。使用 WPF，既可以创建独立的应用程序，也可以创建基于浏览器运行的应用程序。

WPF 的核心是一个与分辨率无关并且基于向量的呈现引擎，能充分利用现代图像硬件的优势。

WPF 是基于现有的 Microsoft .NET Framework 2.0 版本，并与以下 3 项新技术一起构成了 Microsoft .NET Framework 3.0 版本（原来的 WinFX）的部分发布内容，如图 1-2 所示。

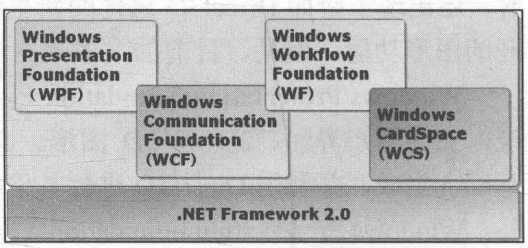


图 1-2　WPF 和其他 Foundation 的关系

- Windows Communication Foundation (WCF)：该项技术可以让开发者生成并运行互联系统。
- Windows Workflow Foundation (WF)：该项技术可以让开发者更加方便地将工作流以及进程添加到应用程序中。
- Windows CardSpace (WCS)身份选择器：该项技术可以让用户简单、安全、信任地使用他们的数字身份进行在线服务。

通过使用.NET Framework 3.0 版本中这些技术，可以创建各种不同类型的应用程序，比如使用 WPF 界面创建单用户系统和大型企业系统，使用 WCF 创建跨境通信系统，使用 WF 创建声明式业务流程，或者使用 WCS 创建在线信息管理程序。

你曾经使用过哪些其他的用户界面框架呢？

1.1.2　WPF 体系结构

WPF 是通过基于.NET Framework 2.0 版本的托管代码公开其编程模型的。WPF 包含下列 3 个主要组件（如图 1-3 所示）：

- PresentationFramework；
- PresentationCore；
- milcore。

其中，PresentationFramework 和 PresentationCore 都是托管组件，而 milcore 是非托管组件。Microsoft DirectX 应用程序接口（API）能够启用 WPF 所有的显示处理，而且提供充足的软硬件渲染。由于 milcore 组件是以非托管代码编写的，所以它可以提供更好的性能，而且能够与 DirectX 引擎进行更为紧密的集成。

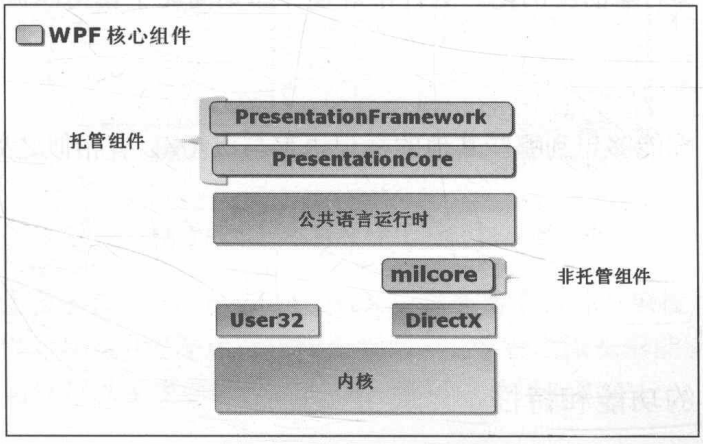


图 1-3　WPF 核心组件

PresentationFramework 和 PresentationCore 能够提供所有在应用程序中用得到的类和 API。

1.1.3　在 WPF 中定义用户界面

可以使用可扩展应用程序标记语言（XAML）中的声明代码生成用户界面，也可以在 Microsoft Visual C#或 Microsoft Visual Basic 中编写命令式代码来生成用户界面。

在大多数情况下，采用这两种方法形成的结果都是相同的，但是使用 XAML 更加易于编写和维护。对于用户界面来说，没有什么不能在 XAML 和代码中完成的。

XAML 是一种用于声明性应用程序编程的标记语言。可以用 XAML 创建自己的用户界面定义，并将应用程序的 XAML 保存在后缀名为.xaml 的文件中。

图 1-4 显示了一段.xaml 文件的代码及其执行效果。

利用 XAML，通过使用.NET Framework 编程模型可以使得创建用户界面（UI）更加方便，因为只需要在声明性 XAML 标记中创建用户界面元素，然后将用户界面定义与应用程序逻辑分离开来即可。XAML 直接描述了托管对象的实例。

所有能够在标记语言中完成的事情，都可以在代码中完成。因此，可以不使用 XAML，

而是直接使用代码来创建自己的用户界面。在 Windows 窗体中，用户界面的序列化格式就是代码，但是在 WPF 中却是 XAML。可以将 XAML 简单描述为 WPF 类的序列化格式。

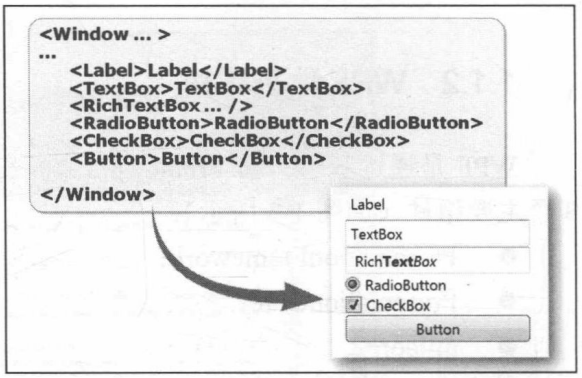


图 1-4　XAML 代码效果

XAML 简化了为 .NET Framework 编程模型创建 UI 的过程。在声明性 XAML 标记中创建可见的 UI 元素，然后使用代码隐藏文件（通过分部类定义与标记相连接）将 UI 定义与运行时逻辑相分离。在 XAML 中混合代码和标记的功能很重要，因为 XML 本身是声明性的，不会为流控制真正建议一个模型。基于 XML 的声明性语言非常直观，可以为用户（尤其是具有 Web 设计和技术背景的人员）创建从原型到生产的各种界面。与其他大多数标记语言不同，XAML 直接呈现托管对象的实例化。这种常规设计原则简化了使用 XAML 创建的对象的代码和调试访问。

你能够想到哪些其他的标记语言与 XAML 有相似之处呢？

1.1.4　WPF 的功能和特性

WPF 作为 .NET Framework 类型的一个子集存在，这些类型大多位于“System.Windows”命名空间。如果用户以前已使用 .NET Framework 通过诸如 ASP.NET 和 Windows 窗体之类的托管技术生成应用程序，那么应该熟悉 WPF 的基本编程体验；可以使用.NET Framework 编程语言（如 C#或 Visual Basic）实例化类、设置属性、调用方法以及处理事件。WPF 提供以下功能和特性：

- 基于 XAML 的用户界面；
- 页面布局管理；
- 数据绑定；
- 2D 和 3D 图形；
- 多媒体；
- 动画；
- 文档和打印；
- 安全；
- 可访问性；
- 本地化；
- 与 Windows 窗体控件的互操作性。

1.1.5　WPF 应用程序的类型

当想要创建传统的基于 Windows 的应用程序，例如文字处理器或者业务应用程序来部署到个体客户端计算机的时候，就可以使用独立应用程序类型。而当想要创建能够在 Web 浏览器中运行的应用程序时，则可以使用 XAML 浏览器应用程序（XBAP）。独立应用程序和 XBAP 的界面如图 1-5 所示。

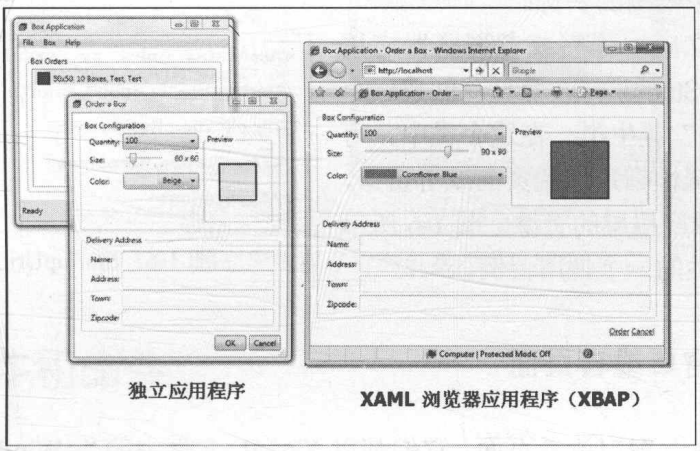


图 1-5　独立应用程序与 XAML 浏览器应用程序

XBAP 更形象地代表了传统 Microsoft ActiveX，或是基于组件对象模型（COM）并承载在浏览器模型中的控件。当应用程序运行权限受到限制，或者当浏览器能够承载该应用程序，或者当主要目的是简化分布以及更新模型的时候，都可以使用 XBAP。

你创建的应用程序中，编写为 XBAP 比编写为独立应用程序哪些更好？

1.2　创建简单的 WPF 应用程序

当想要创建一个 WPF 应用程序的时候，可以使用 Microsoft Visual Studio 2008 开发系统。Visual Studio 2008 提供了大量帮助信息例如项目、文件模版、代码实例、Microsoft IntelliSense 技术、WPF 设计器以及 XAML 编辑器以改进 WPF 开发体验。

本节讲述了当创建一个 WPF 应用程序时，Visual Studio 2008 会生成哪些内容。

1.2.1　演示：使用 Visual Studio 2008 创建 WPF 应用程序

在本演示中，将看到如何：

- 创建独立的 WPF 应用程序；
- 创建浏览器应用程序；
- 添加控件到应用程序。

1.2.2 定义应用程序

Visual Studio 会生成 XAML 应用程序文件，它会指定应用程序的代码后置类、起始窗口或者页面、应用程序范围的资源。

当创建一个独立应用程序或者浏览器应用程序的时候，Visual Studio 2008 会自动创建一个自定义“Application”派生类。在这个类中，可以使用“StartupUri”属性指定起始页面或者窗口，同时也可以指定任意应用层的资源。图 1-6 所示是“StartupUri”属性的一个使用范例。

```
<Application xmlns：x=... xmlns=...
 x：Class="MyApp.App"
 StartupUri="Window1.xaml">

 <Application.Resources>
  ...
 </Application.Resources>

</Application>
```

图 1-6 StartupUri 属性示例代码

1.2.3 定义窗口或者页面

独立应用程序包含窗口或者页面。它们都以 XAML 文件中的“<Window>”或“<Page>”元素表示出来。代码后置文件包含事件处理程序代码。

一个 Window 是最高级别的窗口，而一个 Page 是浏览器承载的页面。WPF 独立应用程序可以包含窗口和页面，但是 WPF 浏览器应用程序只包含页面。图 1-7 所示是“<Window>”和“<Page>”元素的范例代码。

```
<Window xmlns：x=...
xmlns=...
 x:
Class="MyApp.Window1"
 Title="My Window">

 <Grid>
  ...
 </Grid>

</Window>
```

```
<Page xmlns：x=...
xmlns=...
 x：Class="MyApp.Page1"
 WindowTitle="My Page">

 <Grid>
  ...
 </Grid>

</Page>
```

图 1-7 Window 和 Page 元素的范例代码

1.2.4 添加控件

WPF 附带了许多几乎可以在所有 Windows 应用程序中使用的常见 UI 组件，其中包括“Button”、“Label”、“TextBox”、“Menu”和“ListBox”。窗口和页面包含控件，控件会以 XAML 元素表示。这些元素的实例有“<Button>”和“<TextBox>”所有的控件元素，例如“<TextBox>”或“<Button>”都与类相似。可以在独立应用程序和浏览器应用程序中都使用相同的控件。图 1-8 所示是窗口和页面包含控件的范例代码。

```
...
<Grid>
 <TextBox Name="TextBox1" />
 <Button Name="Button1">Click here</Button>
</Grid>
...
```

图 1-8 控件代码范例