TURING 图灵程序设计丛书 .NET系列

SAMS

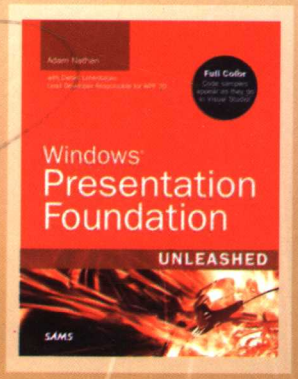
Windows Presentation Foundation Unleashed

WPF揭秘

[美] Adam Nathan 著
翟杰 单佐一 夏寒 译

Amazon
超级畅销书
微软开发部门总经理
强烈推荐

- 通俗易懂，深入实用，揭示大量技术内幕
- 微软.NET核心开发人员力作
- 放飞WPF，赋予你超越梦想的能力



人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 .NET系列

Windows Presentation Foundation Unleashed

WPF揭秘

[美] Adam Nathan 著
瞿杰 单佐一 夏寒 译

人民邮电出版社
北 京

图书在版编目（CIP）数据

WPF 揭秘 /（美）内森（Nathan, A.）著；瞿杰，单佐一，夏寒译. —北京：人民邮电出版社，2008.5
（图灵程序设计丛书）
ISBN 978-7-115-17604-2

Ⅰ. W… Ⅱ. ①内…②瞿…③单…④夏… Ⅲ. 窗口软件，Windows Vista－用户界面－程序设计 Ⅳ. TP316.7

中国版本图书馆 CIP 数据核字（2008）第 017417 号

内 容 提 要

Windows Presentation Foundation (WPF) 是 .NET Framework 3.0 的关键组件，是支持下一代视窗应用程序表现层编程的平台，也是微软新发布的 Vista 操作系统的三大核心开发库之一，主要负责图形显示。本书是针对那些对用户界面开发感兴趣的软件开发人员编写的，易于理解，适合那些 .NET 的新手，并有助于理解像 Microsoft Expression Blend 这样产品的精髓。

本书适合各层次 Web 开发人员阅读。

图灵程序设计丛书

WPF 揭秘

◆ 著 ［美］Adam Nathan
译 瞿 杰 单佐一 夏 寒
责任编辑 陈兴璐

◆ 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
三河市海波印务有限公司印刷
新华书店总店北京发行所经销

◆ 开本：800×1000 1/16
印张：31.75 彩插：4
字数：827 千字 2008 年 5 月第 1 版
印数：1－5 000 册 2008 年 5 月河北第 1 次印刷

著作权合同登记号 图字：01-2007-2673 号
ISBN 978-7-115-17604-2/TP

定价：75.00 元

读者服务热线：(010)88593802 印装质量热线：(010)67129223
反盗版热线：(010)67171154

图4-1　WPF在不同主题下的外观

图4-25　WPF ProgressBar

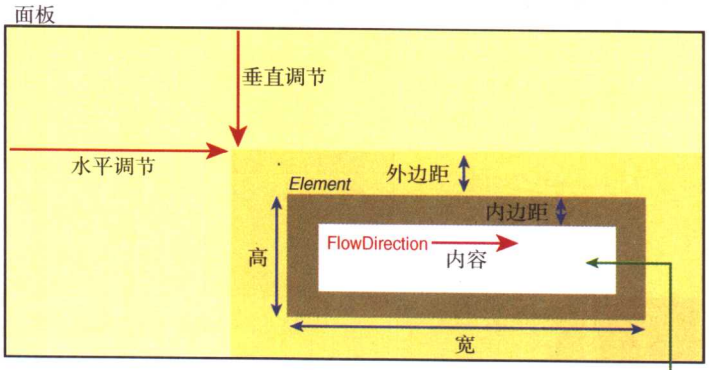


图5-1　本章讲解的主要子布局属性

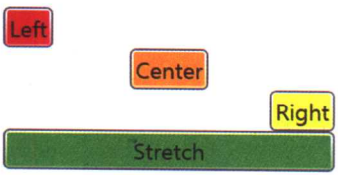
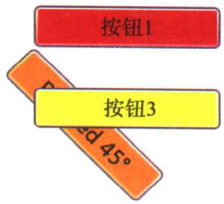


图5-4　在StackPanel中Button上HorizontalAlignment的效果

应用LayoutTransform进行旋转



应用RenderTransform进行旋转

图5-7　LayoutTransform和RenderTransform之间的差别反映在中间那个按钮上

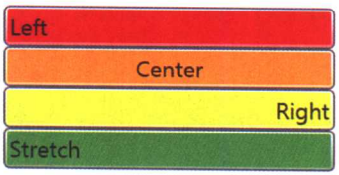


图5-5　StackPanel中Button的HorizontalContentAlignment效果

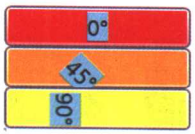


图5-10　内部TextBlock围绕左上角旋转，其中使用了一个醒目的背景

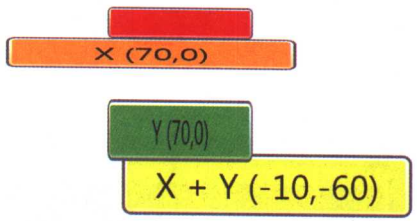


图5-12　代码清单5-2中的按钮，但显式设置了拉伸的中心

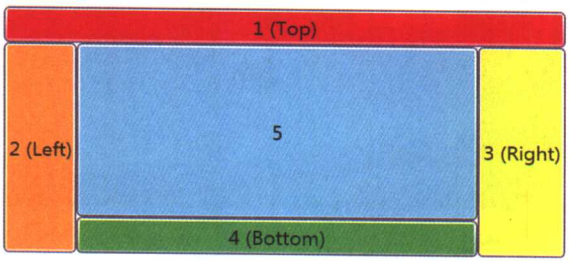


图6-7　一个DockPanel中的按钮

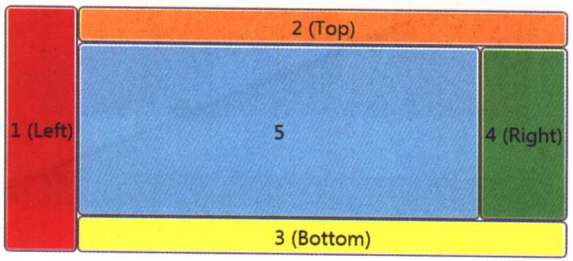


图6-8　一个DockPanel中的按钮，顺序与图6-7不同

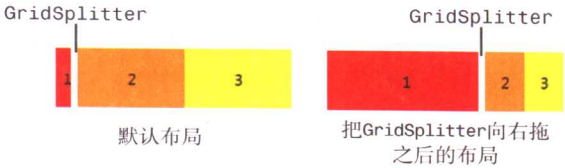


图6-15　一个没有使用SharedSizeGroup的简单Grid

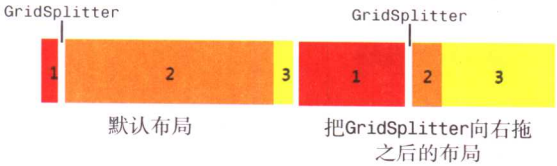


图6-16　图6-15的Grid，但是第1列和最后1列在同一个SharedSizeGroup中

图6-18　ClipToBounds决定了子元素是否呈现在面板之外

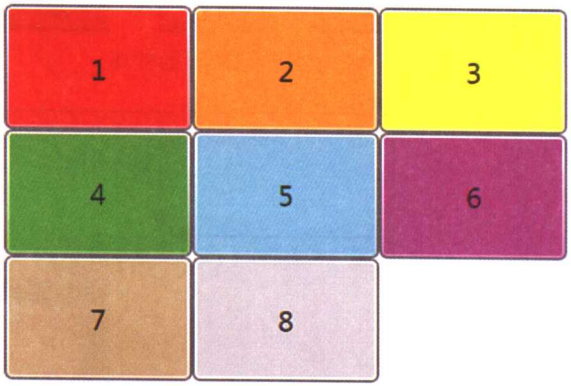


图6-17　添加了8个按钮的UniformGrid

图7-11　整个Window的背景都是玻璃效果

图7-12　只在Window底部扩展玻璃效果

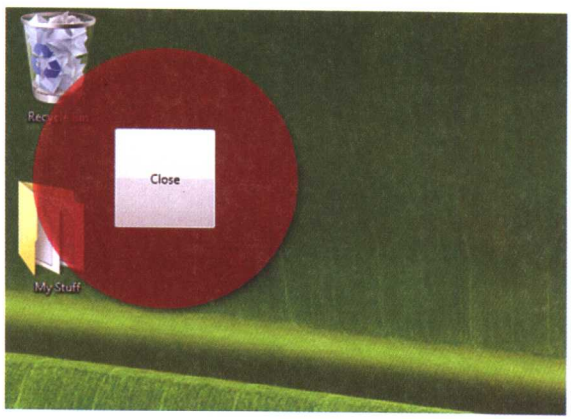


图7-14　一个包含了非矩阵（半透明的）内容的隐藏Window

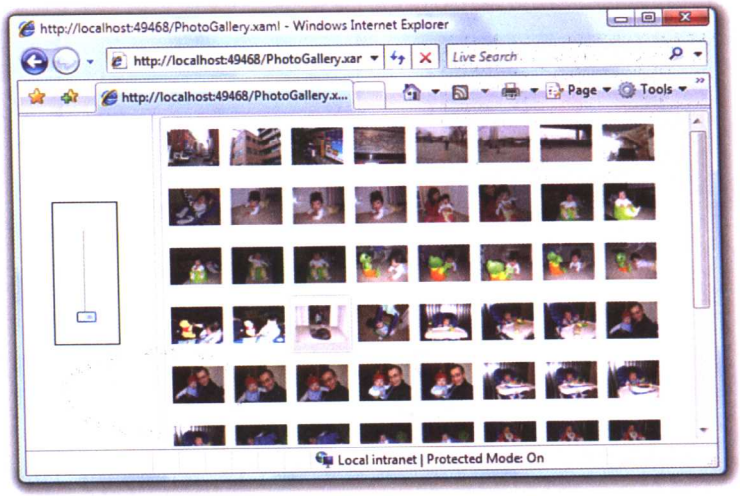


图7-15　Photo Gallery在变成了松散XAML页之后，功能仍然非常强大

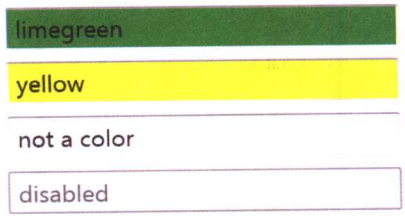


图10-5　当TextBox的字符串为“disabled”时，Style的数据触发器会禁用这个TextBox

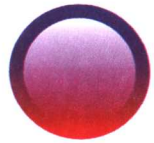


图10-6　由一个自定义的ControlTemplate创建的一个漂亮的圆形Button

图10-7　代码清单10-5中ControlTemplate的悬停和凹下去的特效

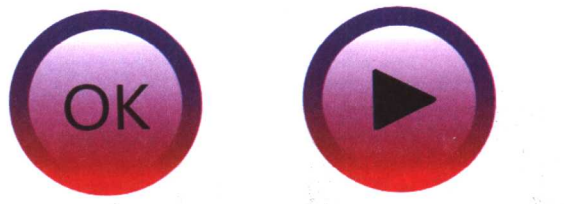


图10-8　两个不同的Button，它们使用了在代码清单10-6中定义的控件模板

图10-9　代码清单10-7中调整了自定义模板外观的Button

图10-11　禁用和不确定的状态下的自定义ProgressBar外观

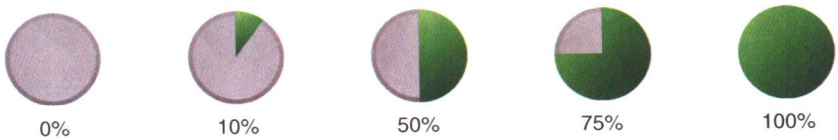
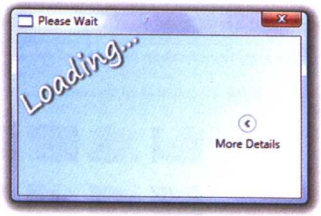


图10-10　不同进度阶段的自定义ProgessBar外观

图10-12　一个对话框，以默认的皮肤显示

“Electric”皮肤



“Light and Fluffy”皮肤

图10-13　对话框的两种皮肤

Windows Vista (Aero)　Windows Classic

图10-14　具有相同Style的相同控件，在两种不同的主题下的外观

Windows Vista (Aero)　Windows Classic

图10-15　在两种不同主题下，应用了主题样式的同一个控件

图11-1　一个简单的EllipseGeometry对象，其中包含了一个GeometryDrawing对象，而GeometryDrawing对象里面有一个DrawingImage对象，DrawingImage对象里面有一个Image对象

图11-14　通过覆盖VisualChildrenCount和GetVisualChild，在Window中绘制鬼怪的DrawingVisual

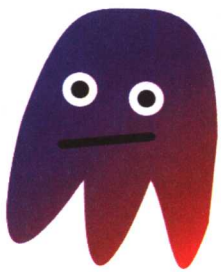
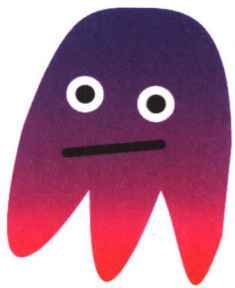


图11-21　应用到鬼怪上的一个简单的蓝红渐变的LinearGradientBrush

SRgbLinearInterpolation

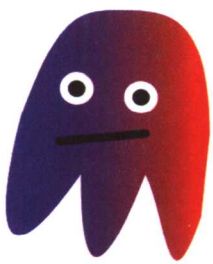


ScRgbLinearInterpolation

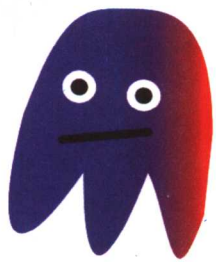
图11-23　ColorInterpolationMode对渐变效果的影响

StartPoint=(0,0), EndPoint=(0,1)

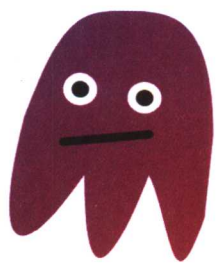
StartPoint=(0,1), EndPoint=(0,0)



StartPoint=(0,0), EndPoint=(1,0)



StartPoint=(0.5,0), EndPoint=(1,0)



StartPoint=(-2,-2), EndPoint=(2,2)

图11-22　不同的SartPoint和EndPoint设置得到的结果

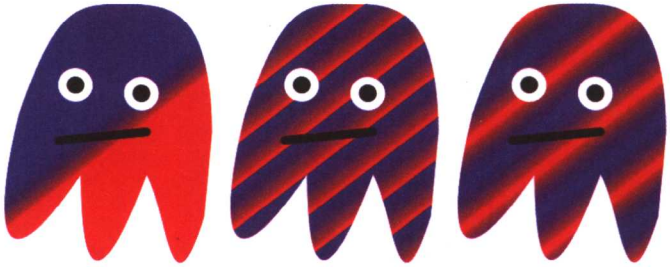


图11-24　不同的SpreadMethod值能产生许多不同的效果

图11-25　使用Pen的LinearGradientBrush画出鬼怪

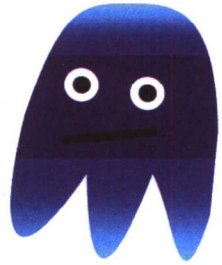


图11-26　通过使用重复的Offset，在渐变上产生两条突变的直线

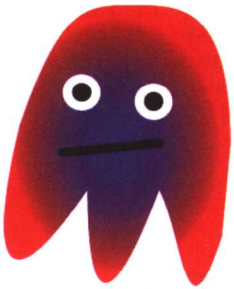


图11-27　应用到鬼怪上的一个简单的蓝红渐变的RadialGradientBrush

图11-28　设置SpreadMetod为Repeat清楚地揭示了椭圆的边界

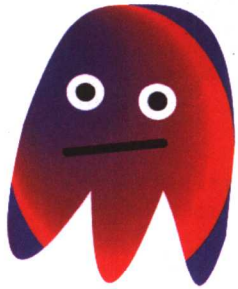


图11-29　用GradientOrigin属性在椭圆内移动渐变的起始点

图11-30　通过使用有透明alpha通道的颜色实现带有半透明效果的鬼怪

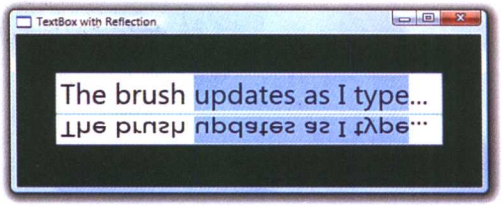


图11-39　一个简单的动态反射效果

图11-40　一个由LinearGradientBrush提供的带条纹状OpacityMask的Button

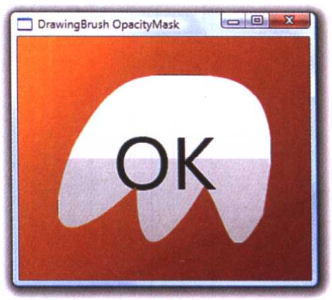


图11-41　将鬼怪作为DrawingBrush OpacityMask，并用两种不同的身体填充色的渲染结果

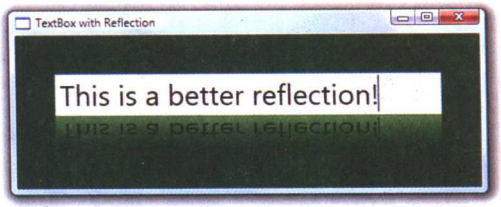


图11-42　用OpacityMask增强后的实时反射效果

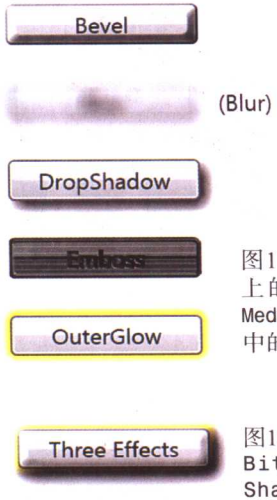


图11-43　应用到Button上的System.Windows.Media.Effects命名空间中的5种位图效果

图11-44　应用到Image的BevelBitmapEffect、DropShadowBitmapEffect和EmbossBitmapEffect

图11-45　3种效果BevelBitmapEffect、DropShadowBitmapEffect和OuterGlowBitmapEffect同时应用到Button上

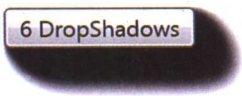


图11-46　6次DropShaowBitmapEffect比5次多太多了

图11-47　设置一些属性可以彻底改变位图效果的外观

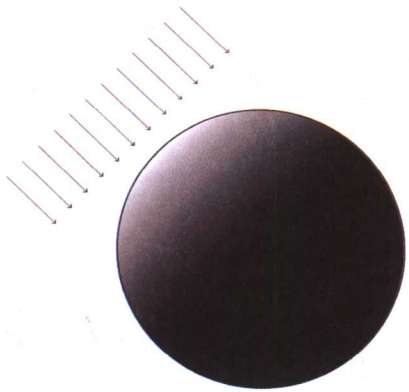


图12-29　DirectionalLight照在一个球体上

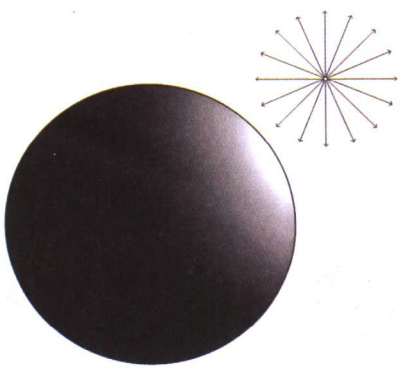


图12-30　PointLight照在一个球体上

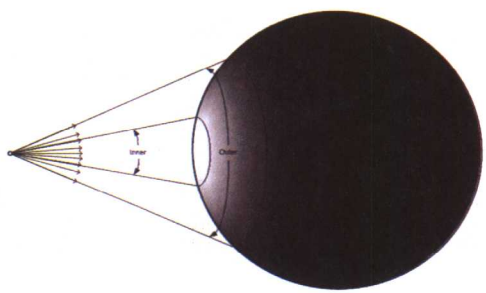


图12-31　SpotLight照在一个球体上

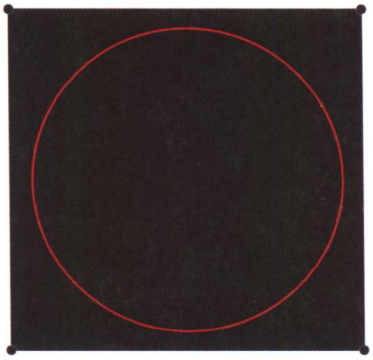


图12-32　四边形内的光线

ImageBrush

色彩为橙色的ImageBrush

图12-39　ImageBrush和色彩为橙色的ImageBrush

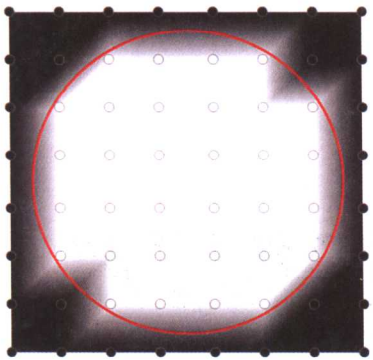


图12-33　在细分网格中的光线

EmissiveMaterial

在黑色DiffuseMaterial上的EmissiveMaterial

图12-40　茶壶模型上的EmissiveMaterial

SpecularMaterial本身

在红色DiffuseMaterial上的SpecularMaterial

图12-41　茶壶模型上的SpecularMaterial

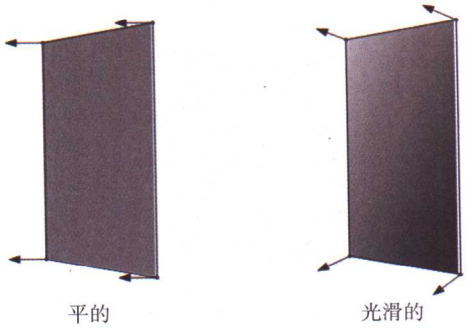


图12-50　使用两个不同Normal值的结果

图13-3　动态改变LinearGradientBrush中间的Color

图13-6　片尾效果的字幕序列快照

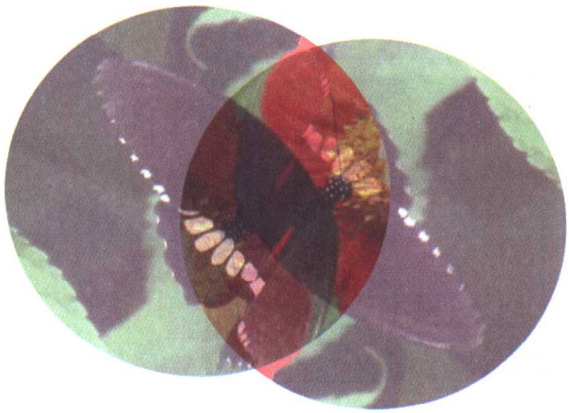


图14-2　在两个MediaElement中经剪辑、旋转的半透明视频

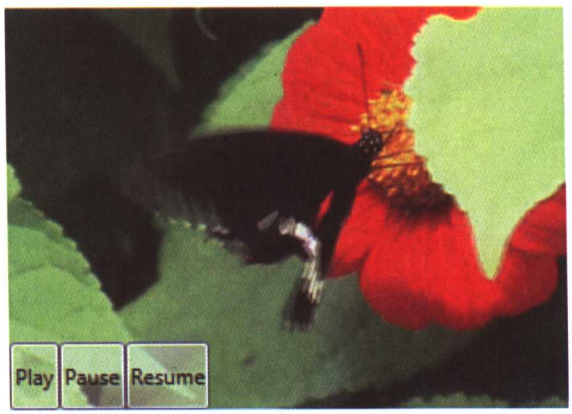


图14-3　一个简单的视频播放器，其中的按钮使用Storyboard来控制视频

版 权 声 明

对本书的赞誉

“要更深入地学习WPF，我强烈推荐Adam Nathan的杰作《WPF揭秘》（Amazon上50位读者授予全五星评价！）。我已经迫不及待读到他的下一部Silverlight著作！”

——Scott Guthrie，ASP.NET之父，微软开发部门总经理

“本书好评如潮，仔细研读之后，果然名副其实。书中的讨论非常深入，而且为继续钻研提供了许多便利，仅凭这一点就远超其他图书……WPF程序员必备。”

——Infoq网站

“我实在想象不出还有什么WPF图书能够超越本书了，作者在透彻讲述技术的意图方面非常出色，而这正是技术图书写作的难点。”

——Patrick Smacchia，微软MVP，《C#和.NET 2.0实战》作者

“本书极为贴近实战，其中透露了大量来自微软公司内部的第一手资料。”

——Ken Cox，微软MVP，*ASP.NET 3.5 For Dummies*一书作者

“这是我读过的最好的WPF图书（至少读了5遍），实例简单明晰。令人惊叹的是，它的内容非常全面，达到了相当的深度和广度，对高级程序员有立竿见影之效。……不可或缺！”

——Amazon.com评语

“真希望更多的软件开发图书能像本书这样出色，充满细节、生动流畅，使读者很快登堂入室。”

——Thomas Duff，资深金融软件程序员

“WPF非常强大，但是要真正高效地运用，却不是容易的事情。这正是本书的价值所在，本书是.NET程序员学习WPF的第一选择。”

——Chris Love，微软MVP

“Adam Nathan的作品可以说开辟了一个新时代，它令人兴趣盎然、欲罢不能，居然使Petzold大师的书相形见绌。”

——Jeff Atwood，资深程序员，著名编程博客（codinghorror.com）

“Adam具有天生的写作天赋，他的上一本讲述.NET与COM的巨著绝版后，众多读者曾经自行发起运动要求出版社重印。本书同样出色，而WPF 3D开发组长Daniel Lehenbauer的加入，更使本书成为该主题不可替代的权威著作。”

——Tim Sneath，微软Silverlight与WPF推广经理

“本书是学习XAML的最佳途径，全书充满激情，我很多年都没有从书中感受到了。”

——Steve C. Orr，微软MVP，*Beginning ASP.NET 2.0 AJAX*一书作者

“本书太棒了！讲得很全面，而且深入，叙述非常生动……如果今年你只想买一本.NET 3.0的书，选择这一本吧。”

——*.NET Developer's Journal*杂志

“与Petzold那本大部头相比，我绝对推荐本书！本书风格截然不同，在我读过的技术图书中，这可能是最赏心悦目的一本。”

——theWPFblog.com

推 荐 序 一

2006年年底，WPF（Windows Presentation Foundation）刚刚随着Windows Vista正式发布的时候，我在中国软件网（http://www.csdn.net）上闲逛，看到两位软件业界资深人士孟岩、韩磊两位老兄针对WPF发布所写的《这一天终于来了》以及《“这一天终于来了”》两篇文章，还有孙辉先生发表的《WPF，一次洗牌……》文章，感慨良多。随着无数业界技术专家在全球范围内的摇旗呐喊，WPF终于开始进入主流视野。

如今事隔一年多，市面上已经有越来越多WPF相关的图书，也有越来越多的开发人员通过搜索引擎、论坛、博客等逐渐熟悉了WPF，WPF已经逐渐成为主流开发选择。但是，仍然有很多朋友对于WPF的定位感到疑惑，毕竟微软在用户交互界面端有太多的新技术。要想真正领会掌握WPF，我们必须先搞明白WPF的定位，也就是为什么微软会推出WPF？

比尔·盖茨在创建微软时曾经有一个梦想——要让每家每户的桌子上都有一台电脑，而且这台电脑上要运行微软的软件。从1975年开始，整个微软就在为这个梦想而奋斗。历史发展到21世纪，应用软件从当初注重性能（硬件的限制）、功能（大而全的软件）而逐步发展到更加注重用户体验。之所以出现这种趋势，是因为软件已经逐渐走下神坛，成为人们日常生活中所不可或缺的东西。而人们对日用品的选择标准就是：价廉物美。价廉导致S+S（Software plus Service）产生，而物美促使软件厂商对于用户体验越来越重视。

正是在这种大势之下，微软才会推出WPF对整个软件生态链催熟。WPF的出现解决了以下3个问题。

(1) 更快速的开发更丰富的用户体验。使用WPF，可以在更短的时间内开发出来更加丰富的界面，以满足用户的需求。WPF的出现，使得我们可以逐渐远离使用控件搭积木开发用户交互界面的开发过程。虽然WPF目前仍然保留控件机制，但我个人认为主要是为了向下兼容。WPF的目标应该是消除控件，让你开发出来的软件根本看不出控件的特征，软件世界再也不是由Button+Textbox组成的怪物了。

(2) 消除用户界面差异。历史在1995年进入一个鸿沟，开发人员突然变成了两大阵营：Browser/Server(B/S)开发人员以及Client/Server(C/S)开发。一个项目启动时，我们首先想到的是这个项目是使用B/S架构还是C/S架构，这两种架构各自有各自的好处，但对于开发人员的知识要求却完全不一样。使用B/S架构，要了解HTML语言以及HTTP协议等；使用C/S，你可能要了解套接字，要了解GDI+等。同样，在项目完成后，如果基于某种原因，需要将此软件架构进行改变。比如将一个C/S项目重新发布为B/S项目，那么对于整个开发团队来说，基本上相当于推倒重来。而现在到了应该弥补这个鸿沟的时候了。WPF正是背负着这个历史责任，同样一套编码，可

以根据你的需要发布C/S架构（Windows Client应用）或者B/S架构（XBAP，XAML Browser Application）。

(3) 软件开发团队的协作问题。软件开发团队的日常协作是一个非常大的问题，除了需求变更以外，第二个影响开发进度的就应该是团队协作性了。在软件团队中，我们比较熟悉开发人员、数据库管理人员、测试人员、运维人员、系统架构师等，而用户交互界面设计师或者说设计人员往往被我们忽视，但实际上，他们的工作成果才是与客户距离最近的。不过就现在的实际情况来说，设计人员与开发人员如何配合工作，是很多团队所头疼的，而WPF正可以解决此问题。因为WPF创造性地引入了XAML语言，开发人员以及设计人员使用这种统一的XML描述的语言进行沟通，将大大降低沟通成本。另外，微软原来为开发人员提供了功能强大的开发工具，也就是Visual Studio系列，而随着WPF的出现，微软也开始面向设计人员提供相应的设计工具：Expression Studio系列。这两套工具所使用的解决方案以及项目结构描述完全一致，也就是完全可以打开对方所创立的项目文件。

如果你浏览一下目前的Windows平台上的软件界面，它们基本上都大同小异，同样都是矩形的窗体，窗体上都是以矩形的控件进行排列。相信已经有很多朋友开始审美疲劳了。不知道有多少朋友在观赏那些好莱坞产的科幻大片时，陶醉于电影主角所使用的更自然更酷的软件交互界面。现在，掌握了WPF，你也能很容易地开发出这种下一代的软件交互界面了。

现在，这本由Adam Nathan所编著由瞿杰等三位译者翻译的《WPF揭密》就摆在你面前，掌握了它，你也就掌握了通往未来的钥匙……

王洪超
互联网策略资深顾问
微软（中国）有限公司
2008/3/3于北京

推 荐 序 二

我第一次接触到WPF（Windows Presentation Foundation）是在MSDN的一段视频上。这段视频演示了如何通过简单的操作来改变控件倾斜角度，轻松地实现一些绚丽的特效。当时，它给我的感觉就是，这将是未来界面设计的方向！

我真正开始使用WPF是在2006年。由于在微软工作的关系，近水楼台先得月，我很早就体验了WPF的beta版。使用之后的切身感受就是，参照一些例子就可以开发小程序，这非常简单，但是想要开发具有出色界面的程序就十分困难——不是 WPF难用，而是不知道怎么用！虽然当时有一些MSDN文章可以参考，但是缺乏系统的讲解。我依葫芦画瓢编写出了XAML，但不知道为什么会得到那种结果，这让我陷入了一种知其然而不知其所以然的窘境。WPF拥有很多新概念，比如“Dependency Properties”、“Routed Events”、“Logical Tree”以及“Visual Tree”等，如果开发人员只会使用这些概念，而没能系统深入地理解它们，必然会受制于它们，更无法开发出高效、绚丽的WPF界面。

2007年，当我准备系统学习WPF的时候，看到这本书，稍稍翻看了几页，顿时眼前一亮，如获至宝。书中把我心中困惑已久的一些问题阐释得清晰明了，还通过很多例子让WPF的学习过程变得非常轻松。我，从此踏上了轻松惬意的 WPF之旅。

可能有人会说MSDN够用了，不需要再看其他的书。我认为这句话只对了一半！MSDN的确是一本百科全书，包罗万象，但是如果你要从头学一门新技术的话，MSDN并不是最合适的。你需要的是一条平坦的学习之路，需要的是一位能由浅入深、循序渐进指引你的导师，那么本书就非常适合。

本书的前几章着重解释WPF的基础技术，第11章～第14章是一个升华——告诉你WPF能开发出如何绚丽的界面，让你每读完一章就有一种跃跃欲试的冲动，甚至每读完一章就会觉得以前应用程序的界面是多么的普通乏味。最后3章是对WPF的一种补充、一种扩展、一种延伸。如果你要真正精通WPF，要成为WPF高手，那么这3章应该是必看的。

在本书的帮助下，我完成了我的一个字典应用作品，参加了微软2007年的“Microsoft酷炫应用争霸赛”，并且幸运地获得了第一名。这第一名无疑就有这本书的功劳，所以当微软的同事瞿杰（Tony Qu）邀请我为其中文版写序的时候，我义不容辞地答应了。

瞿杰为本书的中文版付出了辛勤的劳动。我们经常在回家的路上讨论WPF技术问题，甚至于很多次坐车坐过了站。他对英文术语的中文翻译十分严谨，斟酌再三，尽量使用大家熟悉的中文术语；对于新的术语，也通常会在中文后面带上对应的英文。

当阅读了完稿后的中文版后，我觉得他无数的不眠之夜没有白费。全书保留了英文版的原汁