

Auf einen Blick

Teil I WPF-Grundlagen und Konzepte

1	Einführung in die WPF	41
2	Das Programmiermodell	79
3	XAML	145
4	Der Logical und der Visual Tree	193
5	Controls	237
6	Layout	315
7	Dependency Properties	391
8	Routed Events	431
9	Commands	473

Teil II Fortgeschrittene Techniken

10	Ressourcen	521
11	Styles, Trigger und Templates	569
12	Daten	641

Teil III Reichhaltige Medien und eigene Controls

13	2D-Grafik	781
14	3D-Grafik	857
15	Animationen	903
16	Audio und Video	967
17	Eigene Controls	989
18	Text und Dokumente	1041

Teil IV WPF-Anwendungen und Interoperabilität

19	Windows, Navigation und XBAP	1103
20	Interoperabilität	1163

Inhalt

Vorwort	23
Hinweise zum Buch	27

Teil I WPF-Grundlagen und Konzepte

1 Einführung in die WPF	41
1.1 Die WPF und das .NET Framework	41
1.1.1 Die WPF im .NET Framework 3.0	42
1.1.2 Die WPF und das .NET Framework 3.5	42
1.1.3 Die WPF und das .NET Framework 4.0	43
1.1.4 Die WPF und das .NET Framework 4.5	44
1.1.5 Die WPF als zukünftiges Programmiermodell	46
1.1.6 Stärken und Eigenschaften der WPF	48
1.1.7 Auf Wiedersehen, GDI+	51
1.2 Von Windows 1.0 zur Windows Presentation Foundation	51
1.2.1 Die ersten Wrapper um die Windows-API	51
1.2.2 Windows Forms und GDI+	52
1.2.3 Die Windows Presentation Foundation	53
1.2.4 Die WPF und Windows 8	55
1.3 Die Architektur der WPF	56
1.3.1 MilCore – die »Display Engine«	58
1.3.2 WindowsBase	59
1.3.3 PresentationCore	60
1.3.4 PresentationFramework	60
1.3.5 Vorteile und Stärken der WPF-Architektur	60
1.4 Konzepte	62
1.4.1 XAML	62
1.4.2 Dependency Properties	65
1.4.3 Routed Events	68
1.4.4 Commands	71
1.4.5 Styles und Templates	74
1.4.6 3D	75
1.5 Zusammenfassung	77

2 Das Programmiermodell 79

2.1 Grundlagen der WPF	80
2.1.1 Namespaces	80
2.1.2 Assemblies	81
2.1.3 Die Klassenhierarchie	81
2.2 Projektvorlagen in Visual Studio 2012	88
2.2.1 WPF-Anwendung (Windows)	89
2.2.2 WPF-Browseranwendung (Web)	90
2.2.3 WPF-Benutzersteuerelementbibliothek	91
2.2.4 Benutzerdefinierte WPF-Steuerelementbibliothek	92
2.3 Windows-Projekte mit Visual Studio 2012	93
2.3.1 Ein Windows-Projekt mit XAML und C#	94
2.3.2 Eine reine Codeanwendung (C#)	107
2.3.3 Eine reine, kompilierte XAML-Anwendung	109
2.3.4 Best Practice	112
2.4 Application, Dispatcher und Window	112
2.4.1 Die Klasse »Application«	113
2.4.2 Die Klasse »Dispatcher«	120
2.4.3 Fenster mit der Klasse »Window«	126
2.5 Zusammenfassung	141

3 XAML 145

3.1 <Warum>XAML?</Warum>	145
3.2 Elemente und Attribute	147
3.3 Namespaces	149
3.3.1 Der XML-Namespace der WPF	150
3.3.2 Der XML-Namespace für XAML	152
3.3.3 Über den Namespace-Alias	153
3.3.4 XAML mit eigenen CLR-Namespaces erweitern	154
3.4 Properties in XAML setzen	158
3.4.1 Die Attribut-Syntax	158
3.4.2 Die Property-Element-Syntax	159
3.4.3 Die Content-Property (Default-Property)	161
3.4.4 Die Attached-Property-Syntax	162

3.5	Type-Converter	163
3.5.1	Vordefinierte Type-Converter	164
3.5.2	Eigene Type-Converter implementieren	165
3.5.3	Type-Converter in C# verwenden	169
3.6	Markup-Extensions	171
3.6.1	Verwenden von Markup-Extensions in XAML und C#	171
3.6.2	XAML-Markup-Extensions	174
3.6.3	Markup-Extensions der WPF	175
3.7	XAML-Spracherweiterungen	176
3.8	Collections in XAML	183
3.8.1	Collections, die IList implementieren	183
3.8.2	Collections, die IDictionary implementieren	184
3.9	XamlReader und XamlWriter	187
3.9.1	XAML mit XamlReader dynamisch laden	187
3.9.2	Objekte mit XamlWriter in XAML serialisieren	190
3.10	Zusammenfassung	191

4 Der Logical und der Visual Tree 193

4.1	Zur Veranschaulichung verwendete Komponenten	196
4.1.1	Der InfoDialog von FriendStorage	196
4.1.2	Die Anwendung »XAMLPadExtensionClone«	197
4.2	Der Logical Tree	198
4.2.1	Der Logical Tree des InfoDialogs	199
4.2.2	Für den Logical Tree verantwortliche Klassen	203
4.2.3	Die Klasse LogicalTreeHelper	206
4.2.4	NameScopes, FindName und FindLogicalNode	211
4.3	Der Visual Tree	219
4.3.1	Der Visual Tree des InfoDialogs	219
4.3.2	Eigene Klassen im Visual Tree	222
4.3.3	Die Klasse »VisualTreeHelper«	226
4.3.4	Der Visual Tree und das Rendering	228
4.4	Zusammenfassung	233

5.1	Die Klasse »Control«	240
5.2	ContentControls	242
5.2.1	Buttons	245
5.2.2	Labels	252
5.2.3	ToolTips anzeigen	254
5.2.4	Scrollen mit ScrollViewer	257
5.2.5	WPF- und HTML-Inhalte mit Frame darstellen	259
5.2.6	ContentControls mit Header	261
5.3	ItemsControls	265
5.3.1	ItemsControls mit Header	268
5.3.2	Baumansicht mit der TreeView	271
5.3.3	Navigation über Menüs	276
5.3.4	Elemente mit einem Selector auswählen	280
5.3.5	Das Ribbon	288
5.3.6	Eine StatusBar mit Informationen	292
5.3.7	»Alternating Rows« mit AlternationCount	292
5.4	Controls zur Textdarstellung und -bearbeitung	294
5.4.1	»TextBox« zum Editieren von Text	294
5.4.2	RichTextBox für formatierten Text	296
5.4.3	PasswordBox für maskierten Text	296
5.4.4	TextBlock zur Anzeige von Text	297
5.4.5	Zeichnen mit dem InkCanvas	297
5.5	Datum-Controls	300
5.5.1	Calendar	300
5.5.2	DatePicker	303
5.6	Range-Controls	304
5.6.1	Bereich mit Slider auswählen	305
5.6.2	ProgressBar zur Statusanzeige	306
5.6.3	Scrollen mit der ScrollBar	307
5.7	Sonstige, einfachere Controls	307
5.7.1	Decorator zum Ausschmücken	307
5.7.2	Bilder mit der Image-Klasse darstellen	309
5.7.3	Einfaches Popup anzeigen	310
5.8	Zusammenfassung	313

6.1	Der Layoutprozess	315
6.1.1	Die zwei Schritte des Layoutprozesses	316
6.1.2	»MeasureOverride« und »ArrangeOverride«	318
6.1.3	Ein eigenes Layout-Panel (DiagonalPanel)	319
6.1.4	Zusammenfassung des Layoutprozesses	323
6.2	Layoutfunktionalität von Elementen	325
6.2.1	Width und Height	325
6.2.2	Margin und Padding	326
6.2.3	Alignments	328
6.2.4	Die Visibility-Property	330
6.2.5	Die UseLayoutRounding-Property	331
6.2.6	Transformationen	333
6.3	Panels	344
6.3.1	Die Klasse »Panel«	344
6.3.2	Canvas	346
6.3.3	StackPanel	349
6.3.4	WrapPanel	350
6.3.5	DockPanel	351
6.3.6	Grid	353
6.3.7	Primitive Panels	366
6.3.8	Übersicht der Alignments in den verschiedenen Panels	370
6.3.9	Wenn der Platz im Panel nicht ausreicht	372
6.4	Das Layout von FriendStorage	374
6.4.1	Das Hauptfenster aus Benutzersicht	374
6.4.2	Das Hauptfenster aus Entwicklersicht	376
6.4.3	Animation des Freunde-Explorers	384
6.5	Zusammenfassung	389

7.1	Die Keyplayer	392
7.1.1	»DependencyObject« und »DependencyProperty«	392
7.1.2	Was ist die Property Engine?	394
7.2	Dependency Properties	394
7.2.1	Eine Dependency Property implementieren	396
7.2.2	Metadaten einer Dependency Property	398

7.2.3	Validieren einer Dependency Property	405
7.2.4	Die FontSize-Property als Ziel eines Data Bindings	407
7.2.5	Existierende Dependency Properties verwenden	409
7.2.6	Read-only-Dependency-Properties implementieren	412
7.2.7	Ermittlung des Wertes einer Dependency Property	414
7.2.8	Lokal gesetzte Werte löschen	416
7.2.9	Überblick über die Quellen mit »DependencyPropertyHelper«	417
7.2.10	Auf Änderungen in existierenden Klassen lauschen	417
7.3	Attached Properties	418
7.3.1	Eine Attached Property implementieren	419
7.3.2	Ein einfaches Panel mit Attached Properties	422
7.3.3	Bekannte Vertreter	426
7.4	Zusammenfassung	428

8 Routed Events 431

8.1	Die Keyplayer	432
8.1.1	Die Klassen »RoutedEvent« und »EventManager«	432
8.1.2	Die Routing-Strategie	433
8.1.3	Das Interface »InputElement«	435
8.1.4	Die Klasse »RoutedEventArgs«	437
8.1.5	Das Event System	438
8.2	Eigene Routed Events	439
8.2.1	Ein Routed Event implementieren	439
8.2.2	Das Routed Event als Attached Event verwenden	443
8.2.3	Existierende Routed Events in eigenen Klassen nutzen	445
8.2.4	Instanz- und Klassenbehandlung	447
8.3	Die »RoutedEventArgs« im Detail	452
8.3.1	Sender vs. Source und OriginalSource	452
8.3.2	Die Handled-Property	454
8.4	Routed Events der WPF	456
8.4.1	Tastatur-Events	458
8.4.2	Maus-Events	461
8.4.3	Stylus-Events (Stift)	465
8.4.4	Multitouch-Events	466
8.4.5	Die statischen Mitglieder eines FrameworkElements	469
8.5	Zusammenfassung	470

9	Commands	473
9.1	Die Keyplayer	474
9.1.1	Das Interface »ICommand«	474
9.1.2	Das Interface »ICommandSource«	475
9.2	Eigene Commands mit »ICommand«	476
9.2.1	Ein Command implementieren	476
9.2.2	Das Command verwenden	477
9.2.3	Das Command von der Logik entkoppeln	478
9.3	Die »wahren« Keyplayer	481
9.3.1	Die Klassen »RoutedCommand« und »RoutedUICommand«	481
9.3.2	Der »CommandManager«	483
9.3.3	Die Klasse »CommandBinding«	484
9.3.4	Elemente mit einer CommandBindings-Property	485
9.3.5	Das Zusammenspiel der Keyplayer	486
9.4	Eigene Commands mit der Klasse »RoutedUICommand«	489
9.4.1	Die eigenen Commands in FriendStorage	490
9.4.2	Commands mit »InputGestures« versehen	491
9.4.3	CommandBindings zum Window-Objekt hinzufügen	493
9.4.4	Die Commands im Menü und in der ToolBar verwenden	494
9.5	Built-in-Commands der WPF	499
9.5.1	Built-in-Commands in FriendStorage	500
9.5.2	Bestehende Commands mit InputBindings auslösen	502
9.5.3	Controls mit integrierten CommandBindings	505
9.6	Das Model-View-ViewModel-Pattern (MVVM)	507
9.6.1	Die Idee des Model-View-Controller-Patterns (MVC)	508
9.6.2	Die Idee des Model-View-ViewModel-Patterns (MVVM)	509
9.6.3	Ein MVVM-Beispiel	511
9.7	Zusammenfassung	516

Teil II Fortgeschrittene Techniken

10	Ressourcen	521
10.1	Logische Ressourcen	521
10.1.1	Logische Ressourcen definieren und verwenden	522
10.1.2	Die Suche nach Ressourcen im Detail	525
10.1.3	Elemente als Ressourcen verwenden	529

10.1.4	Statische Ressourcen	532
10.1.5	Dynamische Ressourcen	534
10.1.6	Ressourcen in separate Dateien auslagern	538
10.1.7	Logische Ressourcen in FriendStorage	541
10.2	Binäre Ressourcen	543
10.2.1	Binäre Ressourcen im .NET Framework	544
10.2.2	Binäre Ressourcen bei der WPF	547
10.2.3	Die Pack-URI-Syntax	550
10.2.4	Auf Dateien im Anwendungsverzeichnis zugreifen	551
10.2.5	In C# auf binäre Ressourcen zugreifen	552
10.2.6	Lokalisierung von WPF-Anwendungen	555
10.2.7	Eine binäre Ressource als SplashScreen	565
10.3	Zusammenfassung	568
11	Styles, Trigger und Templates	569
<hr/>		
11.1	Styles	569
11.1.1	Grundlagen und Keyplayer	570
11.1.2	Styles als logische Ressourcen definieren	573
11.1.3	Einen Style für verschiedene Typen verwenden	576
11.1.4	Bestehende Styles erweitern	579
11.1.5	Setter und EventSetter	581
11.1.6	Styles und Trigger	584
11.2	Trigger	584
11.2.1	Property-Trigger	585
11.2.2	DataTrigger	590
11.2.3	EventTrigger	591
11.2.4	Komplexe Bedingungen mit Triggern	596
11.3	Templates	598
11.3.1	Arten von Templates	599
11.3.2	Layout mit dem ItemsPanelTemplate	600
11.3.3	Daten mit DataTemplates visualisieren	601
11.3.4	Das Aussehen von Controls mit ControlTemplates anpassen	605
11.3.5	Das Default-ControlTemplate eines Controls	608
11.3.6	Die Verbindung zwischen Control und Template	612
11.3.7	Two-Way-Contract zwischen Control und Template	615
11.3.8	VisualStateManager statt Trigger verwenden	620
11.3.9	Templates in C#	630

11.4	Styles, Trigger & Templates in FriendStorage	632
11.4.1	Der Next-Button	632
11.4.2	Die Image-Objekte der Toolbar-Buttons	634
11.4.3	Die DataGridRows des Freunde-Explorers	636
11.5	Zusammenfassung	638

12 Daten 641

12.1	Data Binding	642
12.1.1	Data Binding in XAML	642
12.1.2	Data Binding in C#	642
12.1.3	Die Binding-Klasse im Detail	644
12.1.4	Der DataContext	647
12.1.5	Die Path-Property im Detail	649
12.1.6	Die Richtung des Bindings	651
12.1.7	Der UpdateSourceTrigger	653
12.1.8	Die Delay-Property des Bindings	654
12.1.9	Die BindingExpression	655
12.1.10	Bindings entfernen	657
12.1.11	Debugging von Data Bindings	657
12.2	Datenquellen eines Data Bindings	659
12.2.1	Binding an die Dependency Properties eines Elements	660
12.2.2	Binding an einfache .NET-Properties	660
12.2.3	Binding an statische Properties	664
12.2.4	Binding an logische Ressourcen	666
12.2.5	Binding an Quellen unterschiedlichen Typs	667
12.2.6	Binding an relative Quellen mit »RelativeSource«	670
12.2.7	Binding der Target-Property an mehrere Quellen	673
12.2.8	DataSourceProvider für Objekte und XML	677
12.2.9	Binding an X.Linq	684
12.3	Data Binding an Collections	685
12.3.1	Der Fallback-Mechanismus	685
12.3.2	Die CollectionViews der WPF	688
12.3.3	Die DefaultView	692
12.3.4	Daten filtern, sortieren und gruppieren	694
12.3.5	»Live Shaping« von Daten	700
12.3.6	Hinzufügen und Löschen von Daten	701
12.3.7	Collections auf Worker-Threads bearbeiten	703

12.3.8	Mehrere Collections als Datenquelle verwenden	704
12.3.9	Binding an ein ADO.NET-DataSet	705
12.4	Benutzereingaben validieren	708
12.4.1	Validieren mit »ExceptionValidationRule«	710
12.4.2	Validieren mit »DataErrorValidationRule«	712
12.4.3	Validieren mit »NotifyDataErrorValidationRule«	713
12.4.4	Validieren mit eigener ValidationRule	716
12.4.5	Die Validation-Klasse	718
12.4.6	Validieren mehrerer Bindings mit »BindingGroup«	720
12.5	Das DataGrid	729
12.5.1	Die verwendeten Testdaten	730
12.5.2	Autogenerieren von Columns	731
12.5.3	Unterschiedliche Column-Typen	734
12.5.4	Columns manuell zum DataGrid hinzufügen	736
12.5.5	Die Breite einer Column	738
12.5.6	Columns mit der Klasse »DataGridTemplateColumn«	740
12.5.7	RowDetails anzeigen	742
12.5.8	Daten gruppieren	743
12.5.9	Die Auswahlmöglichkeiten festlegen	745
12.5.10	Auf ausgewählte Daten zugreifen	746
12.5.11	Bearbeiten von Daten	747
12.5.12	Daten im DataGrid validieren	748
12.5.13	Sonstige Eigenschaften des DataGrids	752
12.6	Daten mit DataTemplates visualisieren	753
12.6.1	Auswahl mit der Klasse »DataTemplateSelector«	753
12.6.2	Hierarchische DataTemplates	755
12.7	Drag & Drop	758
12.8	Daten in FriendStorage	762
12.8.1	Die Entitäten »Friend«, »Address« und »FriendCollection«	762
12.8.2	Daten im MainWindow	763
12.8.3	Daten im NewFriendDialog	772
12.8.4	Speichern in gezippter .friends-Datei	775
12.9	Zusammenfassung	776

13 2D-Grafik

781

13.1 Shapes	782
13.1.1 Das Rectangle	783
13.1.2 Die Ellipse	784
13.1.3 Linien mit »Line« und »Polyline«	784
13.1.4 Spezielle Formen mit »Polygon«	786
13.1.5 Ein Außerirdischer aus Shapes	789
13.1.6 Die StrokeXXX-Properties der Shape-Klasse	790
13.1.7 Komplexe Shapes mit »Path«	794
13.2 Geometries	794
13.2.1 »RectangleGeometry« und »EllipseGeometry«	795
13.2.2 »LineGeometry«	797
13.2.3 Mehrere Geometry-Objekte gruppieren	797
13.2.4 Geometries kombinieren	798
13.2.5 Komplexe Formen mit »PathGeometry«	799
13.2.6 Die Klasse »StreamGeometry«	802
13.2.7 Die Path-Markup-Syntax	803
13.2.8 Clipping mit Geometry-Objekten	805
13.3 Drawings	806
13.3.1 »GeometryDrawing« und »DrawingGroup«	806
13.3.2 »ImageDrawing« und »VideoDrawing«	808
13.3.3 Ein Außerirdischer aus Geometries und Drawings	809
13.4 Programmierung des Visual Layers	812
13.4.1 Die Klasse »DrawingContext«	812
13.4.2 DrawingVisual einsetzen	814
13.4.3 Visual-Hit-Testing	816
13.5 Brushes	817
13.5.1 Der »SolidColorBrush« und die Color-Struktur	818
13.5.2 Farbverläufe mit GradientBrushes	820
13.5.3 TileBrushes	824
13.6 Cached Compositions	830
13.6.1 BitmapCache für ein Element aktivieren	830
13.6.2 Nebeneffekte des Cachings	831
13.6.3 Elemente mit »BitmapCacheBrush« zeichnen	834

13.7 Effekte	836
13.7.1 Die Effect-Klassen	837
13.7.2 »Blur« und »DropShadow« verwenden	837
13.7.3 Properties von »BlurEffect« und »DropShadowEffect«	839
13.7.4 Effekte mit eigenen Pixelshadern	839
13.7.5 Pixelshader mit weiteren Konstanten	848
13.8 Bitmaps	851
13.8.1 BitmapSources – Bildquellen	851
13.8.2 Bitmap-Operationen	853
13.8.3 Bitmap-Operationen in FriendStorage	853
13.9 Zusammenfassung	854
 14 3D-Grafik	 857
<hr/>	
14.1 3D im Überblick	858
14.1.1 Inhalte einer 3D-Szene	858
14.1.2 2D und 3D im Vergleich	859
14.2 Die Objekte einer 3D-Szene im Detail	861
14.2.1 Das 3D-Koordinatensystem	861
14.2.2 Der Viewport3D als Fernseher	862
14.2.3 Die richtige Kamera	863
14.2.4 Visual3D-Objekte	867
14.2.5 Model3D-Objekte	869
14.2.6 »GeometryModel3D« aufbauen	870
14.2.7 Licht ins Dunkel bringen	876
14.2.8 Transformationen	879
14.2.9 Verschiedene Materialien	881
14.2.10 Texturen	883
14.2.11 Normalen	886
14.3 Benutzerinteraktion mit 3D-Objekten	890
14.3.1 Interaktivität in WPF 3.0 mit Visual-Hit-Testing	890
14.3.2 Interaktivität in WPF 3.5 mit »UIElement3D«	891
14.3.3 Interaktive 2D-Elemente auf 3D-Objekten in WPF 3.5	894
14.4 Komplexe 3D-Objekte	895
14.4.1 Landschaft im Code generieren	896
14.4.2 Kugel erstellen	897
14.4.3 Komplexe 3D-Objekte mit Third-Party-Tools erstellen	899
14.5 Zusammenfassung	900

15.1 Animationsgrundlagen	904
15.1.1 Voraussetzungen für Animationen	904
15.1.2 Übersicht über die Animationsarten und -klassen	905
15.1.3 Timelines und Clocks	908
15.1.4 Das Interface »IAnimatable«	910
15.2 Basis-Animationen in C#	912
15.2.1 Start- und Zielwert mit »From«, »To« und »By«	912
15.2.2 Dauer, Startzeit und Geschwindigkeit	916
15.2.3 Rückwärts und Wiederholen	918
15.2.4 Die Gesamtlänge einer Timeline	919
15.2.5 Wiederholen mit neuen Werten	920
15.2.6 Beschleunigen und Abbremsen	921
15.2.7 Das Füllverhalten einer Animation	923
15.2.8 Eine Animation mit »AnimationClock« steuern	924
15.2.9 Animationen in FriendStorage	928
15.3 Basis-Animationen in XAML	930
15.3.1 Eine einfache Animation in XAML	932
15.3.2 Das Storyboard als Timeline-Container	936
15.3.3 Animationen mit »ControllableStoryboard« steuern	938
15.4 Keyframe-Animationen	940
15.4.1 Lineare Keyframe-Animationen	942
15.4.2 SplineKeyframe-Animationen	944
15.4.3 Animationen mit diskreten Keyframes	945
15.5 Pfad-Animationen	949
15.6 Easing Functions	951
15.6.1 Grundlagen der Easing Functions	951
15.6.2 Easing Functions in Basis-Animationen	955
15.6.3 Easing Functions in Keyframe-Animationen	957
15.6.4 Eigene Easing Functions erstellen	959
15.7 Low-Level-Animationen	961
15.8 Zusammenfassung	964

16.1 Audio (.wav) mit »SoundPlayerAction« und »SoundPlayer«	967
16.1.1 Audio mit »SoundPlayerAction« (XAML)	968
16.1.2 Audio mit »SoundPlayer« (C#)	969
16.2 Audio und Video mit »MediaPlayer« (C#)	971
16.2.1 Einfaches Abspielen	971
16.2.2 Steuerung mit »MediaClock« und »MediaTimeline«	974
16.3 Audio und Video mit »MediaElement« (XAML)	977
16.3.1 Einfaches Abspielen	978
16.3.2 Steuerung mit Methoden (unabhängiger Modus)	979
16.3.3 Steuerung mit »MediaTimeline« (Clock-Modus)	980
16.3.4 Storyboard mit »MediaTimeline« und »AnimationTimeline«	982
16.3.5 Snapshots von Videos	984
16.4 Zusammenfassung	987

17 Eigene Controls

17.1 Custom Controls	990
17.1.1 Die Struktur eines Custom Controls	991
17.1.2 Der zu erstellende VideoPlayer	994
17.1.3 Klassenname anpassen	994
17.1.4 Template-Parts definieren	996
17.1.5 Dependency Properties erstellen	999
17.1.6 Routed Events implementieren	1002
17.1.7 Commands unterstützen	1003
17.1.8 Das Aussehen des lookless Controls festlegen	1007
17.1.9 Das Control testen	1011
17.1.10 Optional weitere Theme-Styles anlegen	1013
17.1.11 Templates auf Windows-Ebene definieren	1016
17.2 Custom Control mit Visual States	1020
17.2.1 Visual States im Code implementieren	1020
17.2.2 States für andere sichtbar machen	1023
17.2.3 States im Default-ControlTemplate unterstützen	1024
17.2.4 Den VideoPlayer mit Visual States testen	1025
17.3 User Control	1026
17.3.1 Die Struktur eines User Controls	1027

17.3.2	Das zu erstellende PrintableFriend-Control	1028
17.3.3	UI des Controls definieren	1029
17.3.4	Properties in der Codebehind-Datei erstellen	1030
17.3.5	Die Content-Property festlegen	1032
17.4	Alternativen zu Custom Control und User Control	1033
17.4.1	Wann sollte man die OnRender-Methode überschreiben?	1033
17.4.2	Adorner erstellen und Elemente damit ausschmücken	1034
17.5	Zusammenfassung	1039

18 Text und Dokumente 1041

18.1	Text	1042
18.1.1	»FrameworkContentElement« als Basis für Text	1042
18.1.2	Formatierung mit Spans	1044
18.1.3	Formatierung mit den Properties aus »TextElement«	1046
18.1.4	Elemente im Text mit »InlineUIContainer«	1048
18.1.5	Fonts und Typefaces	1049
18.1.6	Typografie	1050
18.1.7	Die FormattedText-Klasse	1051
18.1.8	Texteffekte	1053
18.1.9	Nützliche Eigenschaften der TextBlock-Klasse	1054
18.2	Das Text-Rendering beeinflussen	1056
18.2.1	Kleine Zeichen sind schlecht lesbar	1057
18.2.2	Die Schrift führt beim Animieren zu Performance-Problemen	1058
18.2.3	Der Algorithmus für das Anti-Aliasing lässt sich nicht festlegen	1058
18.2.4	Der ClearType-Algorithmus greift nicht immer	1059
18.3	Flow-Dokumente	1061
18.3.1	Die Klasse »FlowDocument«	1061
18.3.2	Die fünf Block-Arten	1063
18.3.3	Die AnchoredBlocks »Figure« und »Floater«	1067
18.3.4	Controls zum Betrachten	1070
18.4	Annotationen	1072
18.5	XPS-Dokumente (Fixed-Dokumente)	1076
18.5.1	FlowDocument als XPS speichern	1077
18.5.2	Ein XPS-Dokument laden und anzeigen	1081
18.5.3	Die Inhalte eines XPS-Dokuments	1082
18.5.4	XPS in C# mit FixedDocument & Co. erstellen	1086

18.6 Drucken	1087
18.6.1 Einfaches Ausdrucken	1088
18.6.2 Drucken mit »PrintQueue«	1090
18.6.3 Festlegen von Druckeigenschaften mit »PrintTicket«	1090
18.6.4 Drucken mit »PrintDialog«	1091
18.7 Dokumente in FriendStorage	1092
18.7.1 Hilfe mit Flow-Dokument	1092
18.7.2 Export der Freundesliste als XPS	1094
18.7.3 Drucken der Freundesliste	1098
18.8 Zusammenfassung	1098

Teil IV WPF-Anwendungen und Interoperabilität

19 Windows, Navigation und XBAP 1103

19.1 Windows-Anwendungen	1104
19.1.1 Built-in-Dialoge	1104
19.1.2 Anwendungen mit UI Automation automatisieren	1106
19.1.3 Deployment	1121
19.2 Windows-Anwendungen und die Windows Taskbar	1122
19.2.1 Übersicht der Möglichkeiten	1123
19.2.2 Thumb-Buttons im Vorschaufenster	1124
19.2.3 Ein Overlay-Bild auf dem Taskbar-Button	1127
19.2.4 Eine Fortschrittsanzeige auf dem Taskbar-Button	1129
19.2.5 Den Ausschnitt im Thumbnail festlegen	1130
19.2.6 Eine JumpList mit JumpTasks	1132
19.2.7 JumpList mit JumpTasks und JumpPaths	1133
19.2.8 JumpList mit letzten und häufigen Elementen	1135
19.3 Navigationsanwendungen	1136
19.3.1 Container für eine Page	1137
19.3.2 Navigation zu einer Seite/Page	1141
19.3.3 Navigation-Events	1147
19.3.4 Daten übergeben	1149
19.3.5 Daten mittels PageFunction zurückgeben	1153
19.4 XBAP-Anwendungen	1156
19.4.1 FriendViewer als XBAP erstellen	1157
19.4.2 Generierte Dateien	1159
19.4.3 XBAP vs. Loose XAML	1160
19.4.4 XBAP vs. Silverlight	1160
19.5 Zusammenfassung	1161

20.1 Unterstützte Szenarien und Grenzen	1163
20.1.1 Mögliche Interoperabilitätsszenarien	1164
20.1.2 Grenzen und Einschränkungen	1165
20.2 Windows Forms	1166
20.2.1 Windows Forms in WPF	1166
20.2.2 WPF in Windows Forms	1174
20.2.3 Dialoge	1176
20.3 ActiveX in WPF	1178
20.4 Win32	1181
20.4.1 Win32 in WPF	1181
20.4.2 WPF in Win32	1191
20.4.3 Dialoge	1195
20.4.4 Win32-Nachrichten in WPF abfangen	1200
20.5 Direct3D in WPF	1202
20.5.1 Voraussetzungen und Konfiguration	1203
20.5.2 Die Direct3D-Oberfläche integrieren	1204
20.6 Zusammenfassung	1207
Index	1209