

Auf einen Blick

TEIL I WPF-Grundlagen und Konzepte

1	Einführung in die WPF	43
2	Das Programmiermodell	79
3	XAML	147
4	Der Logical und der Visual Tree	197
5	Controls	245
6	Layout	325
7	Dependency Properties	403
8	Routed Events	443
9	Commands	485

TEIL II Fortgeschrittene Techniken

10	Ressourcen	533
11	Styles, Trigger und Templates	583
12	Daten	657

TEIL III Reichhaltige Medien und eigene Controls

13	2D-Grafik	801
14	3D-Grafik	877
15	Animationen	923
16	Audio und Video	987
17	Eigene Controls	1009
18	Text und Dokumente	1063

TEIL IV Interoperabilität und Apps

19	Standard-Dialoge, Windows Taskbar und mehr	1127
20	Interoperabilität	1147
21	Universal Windows Apps in Windows 10	1195

Inhalt

Vorwort	27
Hinweise zum Buch	31

TEIL I WPF-Grundlagen und Konzepte

1 Einführung in die WPF	43
1.1 Die WPF und das .NET Framework	43
1.1.1 Die WPF im .NET Framework 4.6	43
1.1.2 Versionen von .NET, C# und Visual Studio	45
1.1.3 Die WPF als zukünftiges Programmiermodell	47
1.1.4 Stärken und Eigenschaften der WPF	49
1.1.5 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	52
1.2.2 Windows Forms und GDI+	52
1.2.3 Die Windows Presentation Foundation	53
1.3 Die Architektur der WPF	55
1.3.1 MilCore – die »Display Engine«	57
1.3.2 WindowsBase	58
1.3.3 PresentationCore	59
1.3.4 PresentationFramework	59
1.3.5 Vorteile und Stärken der WPF-Architektur	59
1.4 Konzepte	61
1.4.1 XAML	61
1.4.2 Dependency Properties	64
1.4.3 Routed Events	67
1.4.4 Commands	71
1.4.5 Styles und Templates	73
1.4.6 3D	75
1.5 Zusammenfassung	76

2.1 Grundlagen der WPF	79
2.1.1 Namespaces	80
2.1.2 Assemblies	80
2.1.3 Die Klassenhierarchie	81
2.1.4 Object	82
2.1.5 DispatcherObject	82
2.1.6 DependencyObject	83
2.1.7 Visual	84
2.1.8 UIElement	84
2.1.9 FrameworkElement	84
2.1.10 Control	86
2.1.11 ContentElement	86
2.1.12 FrameworkContentElement	86
2.1.13 Freezable	87
2.1.14 Visual3D	88
2.1.15 UIElement3D	88
2.2 Projektvorlagen in Visual Studio 2015	88
2.2.1 WPF-Anwendung (Windows)	89
2.2.2 WPF-Browseranwendung (Web)	91
2.2.3 WPF-Benutzersteuerelementbibliothek	92
2.2.4 Benutzerdefinierte WPF-Steuerelementbibliothek	92
2.3 Windows-Projekte mit Visual Studio 2015	94
2.3.1 Ein Windows-Projekt mit XAML und C#	94
2.3.2 MainWindow.xaml	97
2.3.3 MainWindow.xaml.cs (Codebehind)	98
2.3.4 MainWindow.g.cs	99
2.3.5 MainWindow.baml	101
2.3.6 App.xaml	102
2.3.7 App.xaml.cs (Codebehind)	103
2.3.8 App.g.cs	104
2.3.9 Fazit aus den betrachteten Dateien	105
2.3.10 Buildvorgang und MSBuild	105
2.3.11 Eine reine Codeanwendung (C#)	108
2.3.12 Eine reine, kompilierte XAML-Anwendung	111
2.3.13 Best Practice	113
2.4 Application, Dispatcher und Window	114
2.4.1 Die Klasse »Application«	115
2.4.2 Übersicht der Application-Events	115

2.4.3	Events der Application-Klasse verwenden	116
2.4.4	Die Properties »Windows« und »MainWindow«	118
2.4.5	Das Shutdown-Verhalten Ihrer Applikation	120
2.4.6	Die Klasse »Dispatcher«	122
2.4.7	Fenster mit der Klasse »Window«	128
2.4.8	Methoden der Klasse »Window«	129
2.4.9	Die Komponenten eines Fensters	130
2.4.10	Übersicht der Properties der Klasse »Window«	131
2.4.11	Dialogspezifische Properties	137
2.4.12	Modale Dialoge	137
2.4.13	Nicht-modale Dialoge	141
2.4.14	Events der Klasse »Window«	142
2.5	Zusammenfassung	143

3 XAML 147

3.1	<Warum>XAML?</Warum>	147
3.2	Elemente und Attribute	149
3.3	Namespaces	151
3.3.1	Der XML-Namespace der WPF	152
3.3.2	Der XML-Namespace für XAML	154
3.3.3	Über den Namespace-Alias	155
3.3.4	XAML mit eigenen CLR-Namespaces erweitern	156
3.4	Properties in XAML setzen	160
3.4.1	Die Attribut-Syntax	160
3.4.2	Die Property-Element-Syntax	161
3.4.3	Die Content-Property (Default-Property)	163
3.4.4	Die Attached-Property-Syntax	164
3.5	Type-Converter	165
3.5.1	Vordefinierte Type-Converter	166
3.5.2	Eigene Type-Converter implementieren	167
3.5.3	Type-Converter in C# verwenden	172
3.6	Markup-Extensions	173
3.6.1	Verwenden von Markup-Extensions in XAML und C#	174
3.6.2	XAML-Markup-Extensions	176
3.6.3	Markup-Extensions der WPF	178
3.7	XAML-Spracherweiterungen	179

3.8	Collections in XAML	185
3.8.1	Collections, die IList implementieren	186
3.8.2	Collections, die IDictionary implementieren	187
3.9	XamlReader und XamlWriter	189
3.9.1	XAML mit XamlReader dynamisch laden	190
3.9.2	Objekte mit XamlWriter in XAML serialisieren	192
3.10	Zusammenfassung	193

4 Der Logical und der Visual Tree 197

4.1	Zur Veranschaulichung verwendete Komponenten	200
4.1.1	Der InfoDialog von FriendStorage	200
4.1.2	Die Anwendung »XAMLPadExtensionClone«	201
4.2	Der Logical Tree	202
4.2.1	Der Logical Tree des InfoDialogs	203
4.2.2	Für den Logical Tree verantwortliche Klassen	207
4.2.3	Die Klasse LogicalTreeHelper	210
4.2.4	NameScopes, FindName und FindLogicalNode	216
4.2.5	NameScopes – die Namensbereiche	216
4.2.6	INamespace, FrameworkElement und FrameworkContentElement	220
4.2.7	Die Methode »LogicalTreeHelper.FindLogicalNode«	221
4.3	Der Visual Tree	223
4.3.1	Der Visual Tree des InfoDialogs	224
4.3.2	Eigene Klassen im Visual Tree	227
4.3.3	Die Klasse »VisualTreeHelper«	230
4.3.4	Der Visual Tree und das Rendering	233
4.4	Der Live Visual Tree in Visual Studio 2015	238
4.4.1	Live Visual Tree beim Debuggen anzeigen	239
4.4.2	Elemente in der Anwendung auswählen	240
4.4.3	Eigenschaften zur Laufzeit ansehen und anpassen	241
4.5	Zusammenfassung	243

5 Controls 245

5.1	Die Klasse »Control«	248
5.2	ContentControls	250

5.2.1	Buttons	253
5.2.2	Der »normale« Button	253
5.2.3	Der RepeatButton	255
5.2.4	Der ToggleButton	256
5.2.5	Der RadioButton	257
5.2.6	Die CheckBox	259
5.2.7	Labels	260
5.2.8	ToolTips anzeigen	262
5.2.9	Die Klasse »ToolTip«	263
5.2.10	Die Klasse »ToolTipService«	264
5.2.11	Scrollen mit ScrollViewer	265
5.2.12	WPF- und HTML-Inhalte mit Frame darstellen	268
5.2.13	ContentControls mit Header	269
5.2.14	Inhalte mit dem Expander auf- und zuklappen	270
5.2.15	Inhalte mit der GroupBox gruppieren	271
5.2.16	TabItems für ein TabControl	272
5.3	ItemsControls	273
5.3.1	ItemsControls mit Header	275
5.3.2	Navigation mit einerToolBar	276
5.3.3	TreeViewItem und MenuItem	279
5.3.4	Baumansicht mit der TreeView	279
5.3.5	Navigation über Menüs	284
5.3.6	Das gewöhnliche Menü	284
5.3.7	Das Kontextmenü	286
5.3.8	Elemente mit einem Selector auswählen	288
5.3.9	Listen mit der ComboBox anzeigen	289
5.3.10	Eine Liste in der ListBox anzeigen	292
5.3.11	Die ListView als erweiterte ListBox	293
5.3.12	Mehrere Tabs mit einem TabControl	296
5.3.13	Das Ribbon	296
5.3.14	Eine StatusBar mit Informationen	300
5.3.15	»Alternating Rows« mit AlternationCount	301
5.4	Controls zur Textdarstellung und -bearbeitung	302
5.4.1	»TextBox« zum Editieren von Text	303
5.4.2	RichTextBox für formatierten Text	304
5.4.3	PasswordBox für maskierten Text	305
5.4.4	TextBlock zur Anzeige von Text	306
5.4.5	Zeichnen mit dem InkCanvas	306
5.5	Datum-Controls	308
5.5.1	Calendar	309

5.5.2	DatePicker	311
5.6	Range-Controls	313
5.6.1	Bereich mit Slider auswählen	314
5.6.2	ProgressBar zur Statusanzeige	315
5.6.3	Scrollen mit der ScrollBar	315
5.7	Sonstige, einfachere Controls	316
5.7.1	Decorator zum Ausschmücken	316
5.7.2	Einfacher Rahmen mit der Border-Klasse	316
5.7.3	Elemente mit der Viewbox skalieren	317
5.7.4	Bilder mit der Image-Klasse darstellen	318
5.7.5	Einfaches Popup anzeigen	319
5.8	Zusammenfassung	322
6	Layout	325
<hr/>		
6.1	Der Layoutprozess	325
6.1.1	Die zwei Schritte des Layoutprozesses	326
6.1.2	Schritt 1: Measure	326
6.1.3	Schritt 2: Arrange	327
6.1.4	Fazit aus den beiden Schritten	327
6.1.5	»MeasureOverride« und »ArrangeOverride«	328
6.1.6	Ein eigenes Layout-Panel (DiagonalPanel)	329
6.1.7	Zusammenfassung des Layoutprozesses	334
6.2	Layoutfunktionalität von Elementen	335
6.2.1	Width und Height	335
6.2.2	Margin und Padding	336
6.2.3	Margin	336
6.2.4	Padding	337
6.2.5	Alignments	338
6.2.6	»HorizontalAlignment« und »VerticalAlignment«	338
6.2.7	»HorizontalContentAlignment« und »VerticalContentAlignment«	340
6.2.8	Die Visibility-Property	340
6.2.9	Die UseLayoutRounding-Property	341
6.2.10	Transformationen	343
6.2.11	»LayoutTransform« vs. »RenderTransform«	344
6.2.12	»RotateTransform«	347
6.2.13	»ScaleTransform«	348
6.2.14	»SkewTransform«	350

6.2.15	»TranslateTransform«	350
6.2.16	»MatrixTransform«	351
6.2.17	»TransformGroup«	353
6.3	Panels	354
6.3.1	Die Klasse »Panel«	355
6.3.2	Canvas	356
6.3.3	StackPanel	359
6.3.4	WrapPanel	361
6.3.5	DockPanel	362
6.3.6	Grid	364
6.3.7	Die Höhe von Zeilen und die Breite von Spalten	366
6.3.8	GridSplitter	369
6.3.9	Geteilte Größen – »SharedSizeGroup«	373
6.3.10	Das Grid als Alleskönner	375
6.3.11	Primitive Panels	378
6.3.12	»UniformGrid«	378
6.3.13	»VirtualizingPanels«	379
6.3.14	TabPanel	380
6.3.15	ToolBarOverflowPanel	381
6.3.16	Übersicht der Alignments in den verschiedenen Panels	381
6.3.17	Wenn der Platz im Panel nicht ausreicht	383
6.4	Das Layout von FriendStorage	385
6.4.1	Das Hauptfenster aus Benutzersicht	385
6.4.2	Das Hauptfenster aus Entwicklersicht	388
6.4.3	Animation des Freunde-Explorers	396
6.5	Zusammenfassung	401

7 Dependency Properties 403

7.1	Die Keyplayer	404
7.1.1	»DependencyObject« und »DependencyProperty«	404
7.1.2	Was ist die Property Engine?	406
7.2	Dependency Properties	407
7.2.1	Eine Dependency Property implementieren	408
7.2.2	Klassische FontSize-Property	408
7.2.3	»FontSize« als Dependency Property	408
7.2.4	Metadaten einer Dependency Property	410
7.2.5	Die Callback-Methoden der Metadaten	414

7.2.6	Metadaten in Subklassen überschreiben	415
7.2.7	Default-Metadaten	416
7.2.8	Validieren einer Dependency Property	417
7.2.9	Die FontSize-Property als Ziel eines Data Bindings	420
7.2.10	Existierende Dependency Properties verwenden	422
7.2.11	Read-only-Dependency-Properties implementieren	424
7.2.12	Ermittlung des Wertes einer Dependency Property	426
7.2.13	Lokal gesetzte Werte löschen	428
7.2.14	Überblick über die Quellen mit »DependencyPropertyHelper«	430
7.2.15	Auf Änderungen in existierenden Klassen lauschen	430
7.3	Attached Properties	431
7.3.1	Eine Attached Property implementieren	432
7.3.2	Ein einfaches Panel mit Attached Properties	435
7.3.3	Bekannte Vertreter	439
7.4	Zusammenfassung	441
8	Routed Events	443
8.1	Die Keyplayer	444
8.1.1	Die Klassen »RoutedEvent« und »EventManager«	444
8.1.2	Die Routing-Strategie	445
8.1.3	Das Interface »InputElement«	447
8.1.4	Die Klasse »RoutedEventArgs«	449
8.1.5	Das Event System	450
8.2	Eigene Routed Events	450
8.2.1	Ein Routed Event implementieren	451
8.2.2	Klassisches Click-Event	451
8.2.3	Click-Event als Routed Event	452
8.2.4	Das Routed Event als Attached Event verwenden	455
8.2.5	Existierende Routed Events in eigenen Klassen nutzen	457
8.2.6	Instanz- und Klassenbehandlung	459
8.3	Die »RoutedEventArgs« im Detail	463
8.3.1	Sender vs. Source und OriginalSource	464
8.3.2	Die Handled-Property	466
8.4	Routed Events der WPF	468
8.4.1	Tastatur-Events	470
8.4.2	Maus-Events	473
8.4.3	Mouse-Capturing	475

8.4.4	Stylus-Events (Stift)	477
8.4.5	Multitouch-Events	478
8.4.6	Die statischen Mitglieder eines FrameworkElements	481
8.5	Zusammenfassung	482

9 Commands 485

9.1	Die Keyplayer	486
9.1.1	Das Interface »ICommand«	486
9.1.2	Das Interface »ICommandSource«	487
9.2	Eigene Commands mit »ICommand«	488
9.2.1	Ein Command implementieren	488
9.2.2	Das Command verwenden	489
9.2.3	Das Command von der Logik entkoppeln	490
9.3	Die »wahren« Keyplayer	493
9.3.1	Die Klassen »RoutedCommand« und »RoutedUICommand«	493
9.3.2	Der »CommandManager«	495
9.3.3	Die Klasse »CommandBinding«	496
9.3.4	Elemente mit einer CommandBindings-Property	497
9.3.5	Das Zusammenspiel der Keyplayer	498
9.4	Eigene Commands mit der Klasse »RoutedUICommand«	501
9.4.1	Die eigenen Commands in FriendStorage	502
9.4.2	Commands mit »InputGestures« versehen	503
9.4.3	Die Klasse »KeyGesture«	503
9.4.4	Die Klasse »MouseGesture«	504
9.4.5	Die FriendCommands-Klasse mit InputGestures erweitern	504
9.4.6	CommandBindings zum Window-Objekt hinzufügen	505
9.4.7	Die Commands im Menü und in der ToolBar verwenden	507
9.5	Built-in-Commands der WPF	512
9.5.1	Built-in-Commands in FriendStorage	513
9.5.2	Bestehende Commands mit InputBindings auslösen	515
9.5.3	Controls mit integrierten CommandBindings	518
9.6	Das Model-View-ViewModel-Pattern (MVVM)	520
9.6.1	Die Idee des Model-View-Controller-Patterns (MVC)	521
9.6.2	Die Idee des Model-View-ViewModel-Patterns (MVVM)	522
9.6.3	Ein MVVM-Beispiel	524
9.7	Zusammenfassung	529

10 Ressourcen	533
10.1 Logische Ressourcen	533
10.1.1 Logische Ressourcen definieren und verwenden	534
10.1.2 Die Suche nach Ressourcen im Detail	537
10.1.3 Die Suche im Application-Objekt	538
10.1.4 Systemweite Ressourcen	538
10.1.5 Elemente als Ressourcen verwenden	541
10.1.6 Statische Ressourcen	544
10.1.7 Dynamische Ressourcen	546
10.1.8 Ressourcen in separate Dateien auslagern	550
10.1.9 Logische Ressourcen in FriendStorage	553
10.2 Binäre Ressourcen	556
10.2.1 Binäre Ressourcen im .NET Framework	556
10.2.2 Binäre Ressourcen bei der WPF	559
10.2.3 Die Pack-URI-Syntax	562
10.2.4 Auf Dateien im Anwendungsverzeichnis zugreifen	563
10.2.5 In C# auf binäre Ressourcen zugreifen	564
10.2.6 Lokalisierung von WPF-Anwendungen	567
10.2.7 Generelle Lokalisierung mit ResourceManager	568
10.2.8 WPF-Anwendungen mit LocBaml lokalisieren	569
10.2.9 Schritt 1: Default-Kultur für das Projekt definieren	570
10.2.10 Schritt 2: Die Elemente in XAML mit Lokalisierungs-IDs versehen	571
10.2.11 Schritt 3: Weitere Satellite-Assemblies mit LocBaml erstellen	572
10.2.12 Der Test	576
10.2.13 Eine binäre Ressource als SplashScreen	577
10.3 Zusammenfassung	580
11 Styles, Trigger und Templates	583
11.1 Styles	583
11.1.1 Grundlagen und Keyplayer	584
11.1.2 Styles als logische Ressourcen definieren	587
11.1.3 Einen Style für verschiedene Typen verwenden	590
11.1.4 Bestehende Styles erweitern	593
11.1.5 Setter und EventSetter	595
11.1.6 Styles und Trigger	598

11.2	Trigger	598
11.2.1	Property-Trigger	599
11.2.2	Die Reihenfolge der Trigger	601
11.2.3	»EnterActions« und »ExitActions« aus »TriggerBase«	603
11.2.4	DataTrigger	604
11.2.5	EventTrigger	606
11.2.6	»EventTrigger« und »FrameworkElement«	608
11.2.7	Komplexe Bedingungen mit Triggern	611
11.2.8	Logisches Oder	611
11.2.9	Logisches Und	611
11.3	Templates	613
11.3.1	Arten von Templates	613
11.3.2	Layout mit dem ItemsPanelTemplate	614
11.3.3	Daten mit DataTemplates visualisieren	615
11.3.4	Das Aussehen von Controls mit ControlTemplates anpassen	619
11.3.5	Das Default-ControlTemplate eines Controls	622
11.3.6	Die Verbindung zwischen Control und Template	626
11.3.7	Der Inhalt von ContentControls	628
11.3.8	Die Items von ItemsControls	629
11.3.9	Two-Way-Contract zwischen Control und Template	630
11.3.10	VisualStateManager statt Trigger verwenden	634
11.3.11	Ein Template mit Visual States implementieren	636
11.3.12	Zustandsübergänge mit Transitions definieren	639
11.3.13	Das komplette Template im Überblick	641
11.3.14	Templates in C#	645
11.4	Styles, Trigger und Templates in FriendStorage	646
11.4.1	Der Next-Button	646
11.4.2	Die Image-Objekte der Toolbar-Buttons	649
11.4.3	Die DataGridRows des Freunde-Explorers	651
11.5	Zusammenfassung	653

12 Daten 657

12.1	Data Binding	658
12.1.1	Data Binding in XAML	658
12.1.2	Data Binding in C#	658
12.1.3	Die Binding-Klasse im Detail	660
12.1.4	Der DataContext	663
12.1.5	Die Path-Property im Detail	665

12.1.6	Die Richtung des Bindings	667
12.1.7	Der UpdateSourceTrigger	669
12.1.8	Die Delay-Property des Bindings	670
12.1.9	Die BindingExpression	671
12.1.10	Bindings entfernen	673
12.1.11	Debugging von Data Bindings	673
12.2	Datenquellen eines Data Bindings	676
12.2.1	Binding an die Dependency Properties eines Elements	676
12.2.2	Binding an einfache .NET-Properties	677
12.2.3	INotifyPropertyChanged mit C# 6.0	679
12.2.4	»INotifyPropertyChanged« mit dem CallerMemberName-Attribut	681
12.2.5	Binding an statische Properties	681
12.2.6	Binding an logische Ressourcen	684
12.2.7	Binding an Quellen unterschiedlichen Typs	685
12.2.8	Binding an relative Quellen mit »RelativeSource«	688
12.2.9	Mit dem Mode »Self« an eine Property des Target-Elements binden	688
12.2.10	»TemplatedParent« für Templates	689
12.2.11	Mit »FindAncestor« an im Element Tree höher liegende Elemente binden	689
12.2.12	Mit »PreviousData« an das vorhergehende Datenobjekt binden	690
12.2.13	Binding der Target-Property an mehrere Quellen	690
12.2.14	Die Klasse »MultiBinding«	691
12.2.15	Die Klasse »PriorityBinding«	693
12.2.16	DataSourceProvider für Objekte und XML	695
12.2.17	Die Subklasse »ObjectDataProvider«	695
12.2.18	Die Subklasse »XmlDataProvider«	697
12.2.19	Binding an X.Linq	702
12.3	Data Binding an Collections	703
12.3.1	Der Fallback-Mechanismus	703
12.3.2	Die CollectionViews der WPF	706
12.3.3	Das Interface »ICollectionView«	707
12.3.4	Klassen, die »ICollectionView« implementieren	709
12.3.5	Die DefaultView	711
12.3.6	Daten filtern, sortieren und gruppieren	712
12.3.7	Das Filtern, Sortieren und Gruppieren in C#	713
12.3.8	Das Filtern, Sortieren und Gruppieren in XAML	717
12.3.9	»Live Shaping« von Daten	718
12.3.10	Hinzufügen und Löschen von Daten	720
12.3.11	Collections auf Worker-Threads bearbeiten	721
12.3.12	Mehrere Collections als Datenquelle verwenden	722
12.3.13	Binding an ein ADO.NET-DataSet	723

12.4 Benutzereingaben validieren	727
12.4.1 Validieren mit »ExceptionValidationRule«	729
12.4.2 Validieren mit »DataErrorValidationRule«	730
12.4.3 Validieren mit »NotifyDataErrorValidationRule«	732
12.4.4 Validieren mit eigener ValidationRule	735
12.4.5 Die Validation-Klasse	737
12.4.6 Validieren mehrerer Bindings mit »BindingGroup«	739
12.4.7 Validieren mit der BindingGroup	740
12.4.8 Transaktionen mit der BindingGroup	746
12.5 Das DataGrid	748
12.5.1 Die verwendeten Testdaten	749
12.5.2 Autogenerieren von Columns	751
12.5.3 Unterschiedliche Column-Typen	753
12.5.4 Columns manuell zum DataGrid hinzufügen	755
12.5.5 Die Breite einer Column	757
12.5.6 Columns mit der Klasse »DataGridTemplateColumn«	759
12.5.7 RowDetails anzeigen	761
12.5.8 Daten gruppieren	762
12.5.9 Die Auswahlmöglichkeiten festlegen	764
12.5.10 Auf ausgewählte Daten zugreifen	765
12.5.11 Bearbeiten von Daten	766
12.5.12 Daten im DataGrid validieren	767
12.5.13 Sonstige Eigenschaften des DataGrids	771
12.6 Daten mit DataTemplates visualisieren	772
12.6.1 Auswahl mit der Klasse »DataTemplateSelector«	772
12.6.2 Hierarchische DataTemplates	775
12.7 Drag-&-Drop	777
12.8 Daten in FriendStorage	781
12.8.1 Die Entitäten »Friend«, »Address« und »FriendCollection«	782
12.8.2 Daten im MainWindow	782
12.8.3 Der Freunde-Explorer	784
12.8.4 Die Detailansicht	785
12.8.5 Previous/Next-Buttons zur Navigation	786
12.8.6 Data Binding mit Fallback-Mechanismus	787
12.8.7 Validierung des Vornamens	788
12.8.8 Droppen eines Bildes	790
12.8.9 Die StatusBar	791
12.8.10 Daten im NewFriendDialog	792
12.8.11 Speichern in gezippter .friends-Datei	795
12.9 Zusammenfassung	796

13 2D-Grafik	801
13.1 Shapes	802
13.1.1 Das Rectangle	803
13.1.2 Die Ellipse	804
13.1.3 Linien mit »Line« und »Polyline«	804
13.1.4 Spezielle Formen mit »Polygon«	806
13.1.5 Ein Außerirdischer aus Shapes	809
13.1.6 Die StrokeXXX-Properties der Shape-Klasse	810
13.1.7 Komplexe Shapes mit »Path«	813
13.2 Geometries	814
13.2.1 »RectangleGeometry« und »EllipseGeometry«	815
13.2.2 »LineGeometry«	817
13.2.3 Mehrere Geometry-Objekte gruppieren	817
13.2.4 Geometries kombinieren	818
13.2.5 Komplexe Formen mit »PathGeometry«	819
13.2.6 Die Klasse »StreamGeometry«	822
13.2.7 Die Path-Markup-Syntax	823
13.2.8 Clipping mit Geometry-Objekten	825
13.3 Drawings	825
13.3.1 »GeometryDrawing« und »DrawingGroup«	826
13.3.2 »ImageDrawing« und »VideoDrawing«	828
13.3.3 Ein Außerirdischer aus Geometries und Drawings	829
13.4 Programmierung des Visual Layers	832
13.4.1 Die Klasse »DrawingContext«	832
13.4.2 DrawingVisual einsetzen	834
13.4.3 Visual-Hit-Testing	836
13.5 Brushes	837
13.5.1 Der »SolidColorBrush« und die Color-Struktur	838
13.5.2 Farbverläufe mit GradientBrushes	840
13.5.3 TileBrushes	844
13.6 Cached Compositions	849
13.6.1 BitmapCache für ein Element aktivieren	850
13.6.2 Nebeneffekte des Cachings	851
13.6.3 Elemente mit »BitmapCacheBrush« zeichnen	854
13.7 Effekte	856
13.7.1 Die Effect-Klassen	856

13.7.2	»Blur« und »DropShadow« verwenden	857
13.7.3	Properties von »BlurEffect« und »DropShadowEffect«	858
13.7.4	Effekte mit eigenen Pixelshadern	859
13.7.5	Schritt 1: Den Pixelshader erstellen	859
13.7.6	Schritt 2: Den Pixelshader kompilieren	862
13.7.7	Schritt 3: Die .ps-Datei zum Projekt hinzufügen	864
13.7.8	Schritt 4: ShaderEffect-Subklasse erstellen	865
13.7.9	Schritt 5: Den erstellten Effekt einsetzen	867
13.7.10	Pixelshader mit weiteren Konstanten	868
13.8	Bitmaps	871
13.8.1	BitmapSources – Bildquellen	871
13.8.2	Bitmap-Operationen	872
13.8.3	Bitmap-Operationen in FriendStorage	873
13.9	Zusammenfassung	874
14	3D-Grafik	877
14.1	3D im Überblick	878
14.1.1	Inhalte einer 3D-Szene	878
14.1.2	2D und 3D im Vergleich	879
14.2	Die Objekte einer 3D-Szene im Detail	881
14.2.1	Das 3D-Koordinatensystem	881
14.2.2	Der Viewport3D als Fernseher	882
14.2.3	Die richtige Kamera	883
14.2.4	Visual3D-Objekte	888
14.2.5	Model3D-Objekte	889
14.2.6	»GeometryModel3D« aufbauen	890
14.2.7	Licht ins Dunkel bringen	896
14.2.8	Transformationen	899
14.2.9	Verschiedene Materialien	901
14.2.10	Texturen	903
14.2.11	Normalen	906
14.3	Benutzerinteraktion mit 3D-Objekten	910
14.3.1	Interaktivität in WPF 3.0 mit Visual-Hit-Testing	910
14.3.2	Interaktivität in WPF 3.5 mit »UIElement3D«	911
14.3.3	Interaktive 2D-Elemente auf 3D-Objekten in WPF 3.5	913
14.4	Komplexe 3D-Objekte	915
14.4.1	Landschaft im Code generieren	915

14.4.2	Kugel erstellen	917
14.4.3	Komplexe 3D-Objekte mit Third-Party-Tools erstellen	919
14.5	Zusammenfassung	920
15	Animationen	923
15.1	Animationsgrundlagen	924
15.1.1	Voraussetzungen für Animationen	924
15.1.2	Übersicht über die Animationsarten und -klassen	925
15.1.3	Timelines und Clocks	928
15.1.4	Das Interface »IAnimatable«	930
15.2	Basis-Animationen in C#	932
15.2.1	Start- und Zielwert mit »From«, »To« und »By«	932
15.2.2	Animation nur mit »To«	934
15.2.3	Animation nur mit »From«	935
15.2.4	Animation mit »From« und »By«	935
15.2.5	Dauer, Startzeit und Geschwindigkeit	936
15.2.6	Rückwärts und Wiederholen	938
15.2.7	Die Gesamtlänge einer Timeline	940
15.2.8	Wiederholen mit neuen Werten	940
15.2.9	Beschleunigen und Abbremsen	942
15.2.10	Das Füllverhalten einer Animation	943
15.2.11	Eine Animation mit »AnimationClock« steuern	945
15.2.12	Animationen in FriendStorage	949
15.3	Basis-Animationen in XAML	950
15.3.1	Eine einfache Animation in XAML	952
15.3.2	Das Storyboard als Timeline-Container	956
15.3.3	Animationen mit »ControllableStoryboard« steuern	959
15.4	Keyframe-Animationen	961
15.4.1	Lineare Keyframe-Animationen	962
15.4.2	SplineKeyframe-Animationen	965
15.4.3	Animationen mit diskreten Keyframes	966
15.5	Pfad-Animationen	970
15.6	Easing Functions	972
15.6.1	Grundlagen der Easing Functions	972
15.6.2	Easing Functions in Basis-Animationen	976

15.6.3	Easing Functions in Keyframe-Animationen	978
15.6.4	Eigene Easing Functions erstellen	980
15.7	Low-Level-Animationen	982
15.8	Zusammenfassung	985

16 Audio und Video 987

16.1	Audio (.wav) mit »SoundPlayerAction« und »SoundPlayer«	987
16.1.1	Audio mit »SoundPlayerAction« (XAML)	988
16.1.2	Audio mit »SoundPlayer« (C#)	989
16.2	Audio und Video mit »MediaPlayer« (C#)	991
16.2.1	Einfaches Abspielen	991
16.2.2	Audio mit dem MediaPlayer	992
16.2.3	Videos abspielen	992
16.2.4	Steuerung mit »MediaClock« und »MediaTimeline«	994
16.3	Audio und Video mit »MediaElement« (XAML)	998
16.3.1	Einfaches Abspielen	998
16.3.2	Steuerung mit Methoden (unabhängiger Modus)	1000
16.3.3	Steuerung mit »MediaTimeline« (Clock-Modus)	1000
16.3.4	Storyboard mit »MediaTimeline« und »AnimationTimeline«	1003
16.3.5	Snapshots von Videos	1005
16.4	Zusammenfassung	1007

17 Eigene Controls 1009

17.1	Custom Controls	1010
17.1.1	Die Struktur eines Custom Controls	1011
17.1.2	Der zu erstellende VideoPlayer	1013
17.1.3	Klassenname anpassen	1014
17.1.4	Template-Parts definieren	1015
17.1.5	Dependency Properties erstellen	1018
17.1.6	Routed Events implementieren	1022
17.1.7	Commands unterstützen	1023
17.1.8	Das Aussehen des lookless Controls festlegen	1027
17.1.9	Das Control testen	1031
17.1.10	Optional weitere Theme-Styles anlegen	1033

17.1.11	Templates auf Windows-Ebene definieren	1036
17.2	Custom Control mit Visual States	1040
17.2.1	Visual States im Code implementieren	1040
17.2.2	States für andere sichtbar machen	1043
17.2.3	States im Default-ControlTemplate unterstützen	1044
17.2.4	Den MediaPlayer mit Visual States testen	1046
17.3	User Control	1047
17.3.1	Die Struktur eines User Controls	1047
17.3.2	Das zu erstellende PrintableFriend-Control	1049
17.3.3	UI des Controls definieren	1050
17.3.4	Properties in der Codebehind-Datei erstellen	1051
17.3.5	Die Content-Property festlegen	1053
17.4	Alternativen zu Custom Control und User Control	1054
17.4.1	Wann sollte man die OnRender-Methode überschreiben?	1054
17.4.2	Adorner erstellen und Elemente damit ausschmücken	1054
17.5	Zusammenfassung	1060

18 Text und Dokumente 1063

18.1	Text	1064
18.1.1	»FrameworkContentElement« als Basis für Text	1064
18.1.2	Formatierung mit Spans	1066
18.1.3	Formatierung mit den Properties aus »TextElement«	1068
18.1.4	Elemente im Text mit »InlineUIContainer«	1070
18.1.5	Fonts und Typefaces	1071
18.1.6	Typografie	1072
18.1.7	Die FormattedText-Klasse	1073
18.1.8	Texteffekte	1075
18.1.9	Nützliche Eigenschaften der TextBlock-Klasse	1076
18.2	Das Text-Rendering beeinflussen	1078
18.2.1	Kleine Zeichen sind schlecht lesbar	1079
18.2.2	Die Schrift führt beim Animieren zu Performance-Problemen	1080
18.2.3	Der Algorithmus für das Anti-Aliasing lässt sich nicht festlegen	1080
18.2.4	Der ClearType-Algorithmus greift nicht immer	1081
18.3	Flow-Dokumente	1083
18.3.1	Die Klasse »FlowDocument«	1083
18.3.2	Die fünf Block-Arten	1086
18.3.3	Die AnchoredBlocks »Figure« und »Floater«	1090

18.3.4	Controls zum Betrachten	1093
18.4	Annotationen	1095
18.5	XPS-Dokumente (Fixed-Dokumente)	1099
18.5.1	FlowDocument als XPS speichern	1100
18.5.2	Ein XPS-Dokument laden und anzeigen	1103
18.5.3	Die Inhalte eines XPS-Dokuments	1105
18.5.4	XPS in C# mit FixedDocument & Co. erstellen	1109
18.6	Drucken	1111
18.6.1	Einfaches Ausdrucken	1111
18.6.2	Drucken mit »PrintQueue«	1113
18.6.3	Festlegen von Druckeigenschaften mit »PrintTicket«	1114
18.6.4	Drucken mit »PrintDialog«	1114
18.7	Dokumente in FriendStorage	1116
18.7.1	Hilfe mit Flow-Dokument	1116
18.7.2	Export der Freundesliste als XPS	1117
18.7.3	Drucken der Freundesliste	1121
18.8	Zusammenfassung	1122

TEIL IV Interoperabilität und Apps

19 Standard-Dialoge, Windows Taskbar und mehr 1127

19.1	Standard-Dialoge	1127
19.1.1	Der SaveFileDialog	1128
19.1.2	Der OpenFileDialog	1128
19.2	Integration in die Windows Taskbar	1129
19.2.1	Übersicht der Möglichkeiten	1130
19.2.2	Thumb-Buttons im Vorschaufenster	1131
19.2.3	Ein Overlay-Bild auf dem Taskbar-Button	1135
19.2.4	Eine Fortschrittsanzeige auf dem Taskbar-Button	1136
19.2.5	Den Ausschnitt im Thumbnail festlegen	1138
19.2.6	Eine JumpList mit JumpTasks	1139
19.2.7	JumpList mit JumpTasks und JumpPaths	1140
19.2.8	JumpList mit letzten und häufigen Elementen	1142
19.3	Deployment	1144
19.4	Zusammenfassung	1145

20.1	Unterstützte Szenarien und Grenzen	1147
20.1.1	Mögliche Interoperabilitätsszenarien	1148
20.1.2	Grenzen und Einschränkungen	1149
20.2	Windows Forms	1150
20.2.1	Windows Forms in WPF	1150
20.2.2	WPF in Windows Forms	1158
20.2.3	Dialoge	1160
20.2.4	Windows-Forms-Dialoge aus WPF öffnen	1160
20.2.5	WPF-Dialoge aus Windows Forms öffnen	1161
20.3	ActiveX in WPF	1163
20.4	Win32	1165
20.4.1	Win32 in WPF	1165
20.4.2	WPF in Win32	1176
20.4.3	Dialoge	1180
20.4.4	Win32-Dialoge aus WPF öffnen	1180
20.4.5	WPF-Dialoge aus Win32 öffnen	1184
20.4.6	Win32-Nachrichten in WPF abfangen	1185
20.5	Direct3D in WPF	1187
20.5.1	Voraussetzungen und Konfiguration	1188
20.5.2	Die Direct3D-Oberfläche integrieren	1189
20.6	Zusammenfassung	1192

21 Universal Windows Apps in Windows 10

21.1	Einführung	1195
21.1.1	Windows Store Apps in Windows 8	1195
21.1.2	Universal Apps in Windows 8.1	1197
21.1.3	Universal Windows Apps in Windows 10	1198
21.1.4	Universal Windows Apps vs. WPF	1199

21.2 Die FriendViewer-App erstellen	1200
21.2.1 Die Tools installieren	1201
21.2.2 Das Projekt angelegen	1201
21.2.3 Projekt-Struktur und Dateien	1202
21.2.4 Die MainPage befüllen	1204
21.2.5 Visuelle Zustände für adaptives Layout	1209
21.2.6 Wie geht es weiter?	1213
21.3 Zusammenfassung	1213
 Index	 1215