attaching triggers with style, 442
AutoReverse property, 435, 438
BeginStoryboard action, 440
BeginStoryboard.HandoffBehavior property, 443
building gradient-based animations, 474
Canvas as most common layout container for animation, 431
causing animation to repeat itself endlessly, 439
changing 3-D scene using virtual trackball, 923
changing FillBehavior property, 435, 444
code-based animations, 429
ColorAnimation class, 473
comparing key frame animation and sequence of multiple animations, 478
comparing RenderTransform and LayoutTransform, 470
comparing Visibility and Opacity properties, 467
ControllableStoryboardAction class, 445
controlling how animation is repeated, 438
controlling playback, 445
creating 3-D fly-over scene, 921
creating additive animation by setting IsAdditive property, 433
creating animation class for data type, 426
creating animation that fires when window first loads, 441
creating animation that widens button, 430
creating document window that jumps into view, 471
creating dynamic user interfaces, 423
creating event trigger for MouseEnter and MouseLeave events, 521
creating fish-eye effect, 532
creating frame-based animation using nothing but code, 483
creating property trigger that triggers storyboard, 442
CurrentTimeInvalidated event, 451
data types and key frame animation, 426, 479
DataTypeAnimation class, 479
DecelerationRatio property, 438
declarative animation, 439
decreasing frame rate, 461
defining, 26
defining state animations, 575
defining storyboard, 440
defining transform, 468
defining with declarative tags, 5
determining increment size when performing interpolation, 429
determining whether torus mesh has been hit, 926
differentiating from traditional media files, 423
discrete key frame classes, naming format, 478
discrete key frames, 478
displaying position and progress in animation, 450
DoubleAnimation class, 473
DoubleAnimationUsingPath, 482
downloading virtual trackball code, 924
Duration property, 434
easing key frames, 479
EventTrigger.SourceName property, 447
examples of Petzold, Charles, 474
Expression Blend design tool, 472
frame-based animation of falling circles, 484
frame-based animations as not time-dependent, 487
From, To, and Duration properties, 430
fusing second animation into first animation's timeline, 443
future of WPF animation, 498
gradient brushes andRelativeTransform property, 474
Gradient Obsession tool, 474
guidelines for choosing right property to animate, 467
handling Completed event of animation object, 435
handling one-way animations that remain active after finishing, 435
HandoffBehavior.Compose, 443
IAnimatable interface, 429
inheritance hierarchy of WPF animation types, 436
IsCumulative property, 439
key frame animation, definition of, 477

key frame animation, naming format, 426
KeySpline property, 480
LayoutTransform property, 468
linear interpolation, naming format, 426
linear key frames, 478
LinearGradientBrush, 449, 474
managing simultaneous animations as one group, 444
manipulating 3-D objects using transforms, 919
moving camera along route rather than straight line, 921
moving element along path, 427
moving image along path, 482
nesting transforms inside TransformGroup, 471
omitting both From and To properties, 432
omitting From property, 431
omitting To property, 432
path-based animation, naming format, 427
PathGeometry object, 427
performing linear interpolation between key frames, 477
performing wipe transition, 449
Point3DAnimationUsingKeyFrames, 478
PointAnimation class, 473
PointAnimationUsingKeyFrames object, 477
PointAnimationUsingPath class, 483
procedure for creating timer-based animation, 424
progressively revealing element, 468
property-based animation, definition of, 425
RadialGradientBrush, 472, 474
reference types as not usually animated, 426
rendering animation inactive by calling BeginAnimation( ) method, 435
RenderTransform property, 468
RepeatBehavior property, 438
resetting animated element to its original state, 435
reversible, 435
RotateTransform, 468, 472
rotating 3-D object around specified axis, 920, 932
rotating button on mouseover, 469
ScaleTransform, 468, 472
setting RenderTransform property of Border object, 472
setting RenderTransformOrigin property, 469
similarity of Duration property and TimeSpan object, 434
snapshot-and-replace behavior, 443
spline key frame example, 480
spline key frames, naming format, 480
stopping vs. completing animation, 446
storyboard actions and properties, 448
storyboard, definition of, 440
storyboard events, table of, 450
TargetName property, 440
TargetProperty property, 440
Timeline class, table of properties, 437
TimelineGroup class, 437
TranslateTransform, 468
TypeNameAnimationBase class, 427
using animation class that supports dependency property's data type, 426
using BeginAnimation( ) method, 429, 434
using brush properties to animate element's surface, 468
using By property instead of To property, 433
using Canvas to animate position, 467
using discrete key frames in RadialGradientBrush example, 478
using event triggers
  to attach animation, 442
  to control storyboard, 439
using opacity masks, 449
using PathGeometry object to set property, 481
using RepeatBehavior property to set repeat interval, 439
using series of key frames, 478
using transforms to animate element's visual appearance, 468
using Trigger.EnterActions and Trigger.ExitActions, 443
Vector3DAnimationUsingKeyFrames, 478
working with Bézier curves, 480
working with overlapping animations, 443
working with various animation classes for different data types, 428

WPF's standard frame rate, 429
animation easing, 452–457
animation programming, 498
description, 423
easing function classes, 452, 455–457
EasingFunction property, 452–454
animation programming
Completed event, Storyboard class, 487, 496
main page, 488
user controls, 490–491
AnimationTimeline class, 437
Annotation class
properties as read-only, 985
retrieving information from Anchors and Cargos properties, 983
table of properties, 983
Annotation.AnnotationType property, 984
AnnotationDocumentPaginator class, 999
AnnotationHelper class, 977, 979, 985
annotations
accepting hand-drawn ink content in note window, 980
adding comments and highlights to flow/fixed documents, 976
AnnotationHelper class, 977, 979
AnnotationService class, 977
AnnotationStore class, 977
attaching sticky notes to selected text, 976
CreateHighlightCommand, 981
CreateInkStickyNoteCommand, 980
creating, deleting, and highlighting annotations, 979
creating FileStream, 978
customizing appearance of sticky notes, 987
DeleteStickyNotesCommand, 980
enabling Annotation Service, 977
examining and manipulating existing annotations, 985
GetAnnotations( ) method, 982
giving every StickyNoteControl new background color, 987
having multiple users annotate same document, 980
hiding and restoring sticky notes, 979
highlighting text, 976
higlighting content with semitransparent color, 981
mapping System.Windows.Annotations namespace, 979
printing document that includes annotations, 982
procedure for storing annotations in XPS document file, 986
reacting to annotation changes, 986
StickyNoteControl class, 987
storing annotations in MemoryStream, 978
storing position of each note window in AnnotationService, 980
support for, in WPF document containers, 976
System.IO.Packaging namespace, 986
using different control template for StickyNoteControl, 987
XmlStreamStore class, 977
Annotations classes, WPF, 977
AnnotationService class, 977
AnnotationStore class, 977, 982
AnnotationType property, Annotation class, 983
AnnotionService class, 978
antialiasing, 2–3, 9
App_Startup( ) method, 224
AppDomainUnloadedException attribute, 1067
AppendSsml( ) method, 887
AppendText( ) method, 886
AppendTextWithHind( ) method, 886
App.g.cs file, 218
Application class, 798, 808
accessing current application instance, 224
allowing interactions between windows, 225
App_Startup( ) method, 224
App.g.cs file, 218
Application events, table of, 220
App.xaml, 217, 527
App.xaml.cs file, 220
calling base class implementation, 222
casting window object to right type, 225
creating dedicated method in target window, 758
Current property, 224
custom Application class overriding OnSessionEnding, 221
deriving custom class from, 217
DispatcherExceptionUnhandled event, 221

DoUpdate( ) method, 759
examining contents of Application.Windows collection, 225
function of, 215
Main( ) method, 216, 220
MainWindow property, 216, 758
minimizing need for window interactions, 758
one-to-many window interaction, 759
reading command-line arguments, 223
ReasonSessionEnding property, 220
Run( ) method, 216, 220
Shutdown( ) method, 219–220
ShutdownMode property, enumeration values, 218
single window interaction, 759
Startup event, 216
storing references to important windows, 225
Visual Studio and, 217
Application element, 28
application life cycle
application events, 220–222
creating application object, 216
deriving custom application class, 217–218
overview, 215
shutdown, 218–219
application menu, 856, 858
application resources, considering trade-off between complexity and reuse, 300
Application tag, 217
application tasks
handling command-line arguments, 223–224
overview, 222
Application Updates dialog box, 1095
application windows, determining size of, 8
ApplicationCommands class
list of all commands, 270
Undo command, 286
ApplicationCommands.Print command, 997, 1017
ApplicationCommands.Undo command, 555
ApplicationExtension tag, 820
Application.GetResourceStream( ) method, 169
ApplicationName.exe, 822
ApplicationName.exe.manifest, 822
ApplicationName.xbap, 822
ApplicationPath property, JumpTask class, 782
applications
interacting between windows, 225–227
single-instance
Application.Startup event, 228
creating, 227
FileRegistrationHelper class, 232
Microsoft Word, 228
registering file extension using Microsoft.Win32 namespace, 232
ShowDocument( ) method, 229
SingleInstanceApplicationWrapper class, 229
using systemwide mutex, 228
Windows Communication Foundation, 228
WpfApp class, 229
wrapping WPF application with WindowsFormsApplicationBase class, 228
Application.Startup event, 228
Application.StartupUri property, 223
Application.Windows collection, 225
ApplyPropertyValue( ) method, 971
App.xaml, 217, 220, 527, 1029
architecture of WPF
core WPF namespaces, 14
DependencyObject class, 13
Direct3D, 13
dispatcher, 15
DispatcherObject class, 13, 15
elements and controls compared, 16
Media Integration Layer (MIL), 13
milcore.dll, 13
PresentationCore.dll, 13
PresentationFramework.dll, 13
single-thread affinity (STA), 15
UIElement class, 13
User32, 13
Visual class, 13
WindowsBase.dll, 13
WindowsCodecs.dll, 13
ArcSegment class, 384
drawing arcs, 385
Point property, 385

setting IsLargeArc property, 386
Size property, 385
SweepDirection property, 386
Arrange( ) method, 585, 994
arrange stage, 63
ArrangeCore( ) method, 583
ArrangeOverride( ) method, 64, 583, 585–586, 589
asInvoker application, 233
ASP.NET, similarity of tagging syntax to HTML, 25
assembly resources
accessing AssemblyName.g.resources resource stream, 236
adding resources, 234
AssemblyAssociatedContentFile attribute, 239
binary resources, 234
BitmapImage object, 237
Build Action property, 234
ContentType property, 236
definition of, 293
Embedded Resource build action, 235
GetResourceStream( ) method, 236
grouping and organizing, 234
marking noncompiled files as content files, 239
Reflector, 235
resource stream, naming convention, 235
resource-aware classes, 237
ResourceManager class, 236
Resources tab, Project Properties window, 235
ResourceSet class, 236
retrieving, 236
retrieving resources embedded in another library, 238
Stream property, 236
StreamResourceInfo object, 236
syntax in WPF, 237
UnmanagedMemoryStream object, 236–237
using build type of Resource, 235
using disassembler, 235
using strong-named assembly, 238
when deploying resource files isn't practical, 239
working with resources natively, 237
AssemblyAssociatedContentFile attribute, 239
AssemblyName, 46
Asset Library, 330–331
asynchronous printing, 1020
AsyncOperation class, 1050
AsyncOperationManager class, 1050
attached events, 129
attached properties, 68
creating, 112
defining, 112
function of, 112
GetValue( ), 112
setting on any dependency object, 112
SetValue( ), 112
translating into method calls, 38
two-part naming syntax, 38
using to control layout, 38
attributes
Class, 30
ContentProperty, 39–41
Name, 31
RuntimeNameProperty, 32
setting XAML class properties through attributes, 27
TypeConverter, 34
using to attach event handlers, 44
XAML start tag, 28
x:Key, 39
xmlns, 29
xml:space="preserve", 43
audio
audio file with two synchronized animations, code example, 879
inability to retrieve spectrum data in WPF, 865
MediaElement class, 871
playing WAV audio, 865
SoundPlayer class, 866
SoundPlayerAction class, 867
Authors property, Annotation class, 983
AutoComplete feature, 686, 1021
AutoGenerateColumns property, DataGrid, 732, 735
automation, 1071
AutoReverse property, 435, 437–438
AutoWordSelection, 199

axis lines, using as testing tool, 902

B

BackContent property, FlipPanel control, 570
BackEase class
  Amplitude property, 455
  easing function classes, 455
background, 497
Background property, 751, 795, 939
BackgroundWorker class, 1025
BackgroundWorker component, 1045
  adding support for canceling long-running task, 1052
  adding support for tracking progress, 1051
  AsyncOperation class, 1050
  AsyncOperationManager class, 1050
  calling ReportProgresss( ) method, 1051
  cancel messages, 1046
  CancelAsync( ) method, 1052
  creating, 1047
  declaring in XAML, 1047
  DoWork event, 1049
  executing, 1048
  FindPrimes( ) method, 1046
  performing sieve of Eratosthenes algorithm asynchronously, 1046
  progress events, 1046
  ProgressChanged event, 1051
  RunWorkerCompleted event, 1049, 1053
  SynchronizationContext class, 1050
  System.ComponentModel namespace, 1047
  using with single asynchronous background task, 1046
  Window.Resources collection, 1048
  WorkerReportsProgress property, 1051
  WorkerSupportsCancellation property, 1052
BackMaterial property, 895
BackStack property, 811
Balance property, 877
BAML, 26, 48, 53
Band property, 850
BandIndex property, 850
BasedOn attribute, 318
BasedOn property, 313, 719
BeginAnimation( ) method, 429, 434–435
BeginChange( ) method, 200
BeginInit( ) method, 134
BeginInvoke( ) method, 1043–1044
BeginningEdit event, DataGrid, 748
BeginStoryboard action, 440
BeginStoryboard.HandoffBehavior property, 443
BeginStyle( ) method, 886
BeginTime property, TimeLine class, 437
Behavior class
  restricting behavior to specific elements, 327
  reusable behaviors, 327
behavior classes, 327–329
behaviors, 309
  design-time behavior support in Expression Blend, 330
  getting support for, 325
  making elements draggable with, 330
  reusable behaviors, 327
BetweenShowDelay property, ToolTipService, 185
Bevel line join, 349
Bézier curves, 480, 483
BezierSegment class, 384, 387
Bgr32 format, 417
Bgra32 format, 417
binary resources, 234
Binding class, 251, 257
binding expressions, defining DataGrid columns, 736
Binding markup extension, 251, 254
Binding property, 736
BindingExpression class, 260
BindingExpression.UpdateSource( ) method, 259–260
BindingList collection, 614
BindingListCollectionView, 691, 701
BindingMode enumeration, table of values, 252
BindingOperations class, 254
Binding.RelativeSource property, 554, 681, 683
Binding.StringFormat property, 645, 647
Binding.ValidatesOnDataErrors property, 625
Binding.ValidatesOnExceptions property, 625
Binding.ValidationRules collection, 627
Binding.XPath property, 641
bi-pane proportional resizing, lack of in .NET 1.x, 61

bitmap caching, 463–464
bitmap effects
  BitmapEffect class, 411
  goal of, 411
  implementing only in unmanaged code, 412
  rendering in software and not on video card, 412
bitmap scaling, 11
BitmapCacheBrush class, 353
BitmapEffect class, 411
BitmapImage class, 652
BitmapImage object, 237
bitmaps, 416
BitmapSource class, 416
BlackoutDates property, DatePicker class, 211
BlackWhite format, 417
BleedBox property, 997
Blender, 910
block elements
  BlockUIContainer class, 948
  definition of, 939
  List element, 944
  Paragraph element, 943
  Section element, 948
  Table element, 945
  table of formatting properties, 940
Blocks collection, 957
BlockUIContainer class
  example of placing button under paragraph, 949
  placing noncontent elements inside flow document, 948
  placing noncontent elements inside flow documents, 974
BlurEffect class, 411–412
  animating pixel shaders, 476
  Radius property, 412
BlurRadius property, DropShadowEffect class, 413
Bold element, 949
Bomb user control, 490–491, 494
bomb-dropping game
  counting bombs dropped, 496
  dropping bombs, 491–494
  intercepting bombs, 494–496
  main page, 488
  possible refinements, 497–498
  user controls, 490–491
Border class, 74, 514, 591
Border element, 558
BorderBrush, 74
BorderBrush property, 162, 751, 940, 947
BorderThickness, 74
BorderThickness property, 162, 751, 940, 947
Bottom property, 94
BounceEase class, 455
Bounces property, BounceEase class, 455
Bounciness property, BounceEase class, 455
Box value, MarkerStyle property, 944
BringIntoView( ) method, LogicalTreeHelper class, 503
Brush class, 352, 394
Brush object, 160
brushes
  adding more than two GradientStops, 355
  adding one GradientStop for each blended color, 355
  animating, 472
  animating element that's filled with VisualBrush, 474
  animating radial gradient along ellipse, 472
  automatic change notification, 162
  background and foreground, 160
  BorderBrush property, 162
  BorderThickness property, 162
  Brush object, 160
  building gradient-based animations, 474
  ColorAnimation class, 473
  comparing proportionally sized and fixed-sized tiles, 360
  creating gradients with more than two colors, 355
  creating radial gradient with offset center, 357
  deriving from Freezable, 352
  DoubleAnimation class, 473
  gradient brushes and RelativeTransform property, 474
  Gradient Obsession tool, 474
  ImageBrush, 358, 371
  LinearGradientBrush, 162, 354, 474
  markup for button that fades from solid to transparent, 373

markup for shading rectangle diagonally, 354
Opacity property, 352
painting border around controls, 162
PointAnimation class, 473
RadialGradientBrush, 356, 472, 474
setting button's surface color, 160
SolidColorBrush, 161
support for change notification, 352
support for partial transparency, 352
SystemBrushes class, 352
TileBrush, 371
tiling image across surface of brush, 360
using color that has nonopaque alpha value, 370
using tiled ImageBrush, 360
VisualBrush, 363
bubbling, 124, 126, 131, 628
Build Action property, 234
Built-in element, 150
BulletChrome class, 509
Button class
ContentPresenter as included in its control template, 514
IsCancel property, 177
IsDefault property, 177
IsDefaulted property, 178
Button control, 171, 177
Button object, 163
<Button> element, 27
ButtonBase class, 177
ButtonChrome class, 506, 509, 593–594
ButtonChrome decorator, 75, 163
buttons
animating pixel shaders, 475
blurred buttons, 412
ButtonState event, 145
By property, 433
byte arrays, 1064

C

C#
lack of tools for graphic designers, 24
partial classes, 30
Windows Forms and, 24
CacheMode property, 464
caching, bitmap, 463
CAL (Composite Application Library), 1056
CalendarClosed event, DatePicker, 213
CalendarOpened event, DatePicker, 213
caller inform design pattern, 603
camera
3-D animation that flies around torus, 921
animating LookDirection property of, 921
animating UpDirection property of, 922
changing 3-D scene using virtual trackball, 923
creating 3-D fly-over scene, 921
downloading virtual trackball code, 924
moving along route rather than straight line, 921
placing and configuring, 899
setting Viewport3D.Camera property with Camera object, 899
Camera property, 890
CancelAsync( ) method, 1020, 1052
CanContentScroll property, 191
CanExecute( ) method, 268–269, 277, 279
CanExecute property, 279
CanExecuteChanged event, 268, 277, 279
CanExecuteRoutedEventArg class, 279
CanGoBack property, 807, 810
CanGoForward property, 807
CanUndo property, 199
CanUserReorderColumns property, DataGrid, 735
CanUserResize property, DataGrid, 735
CanUserResizeColumns property, DataGrid, 735
CanUserSort property, DataGrid, 746
CanUserSortColumns property, DataGrid, 746
Canvas
animation and, 431
Bottom property, 94
ClipToBounds property, 95, 343, 462
code example with four buttons, 94
controlling layering of overlapping elements, 95
controlling shape placement and overlap, 340
description of, 65
Height property, 94

Left property, 94
lightweight features of, 95
nesting inside user interface, 95
placing elements using exact coordinates, 94
placing Line in Canvas, 344
promoting element by increasing its ZIndex, 96
Right property, 94
SetZIndex( ), 96
tag order and overlapping shapes, 340
Top property, 94
using to animate position, 467
using Viewbox element to resize shapes proportionally, 341
Width property, 94
ZIndex property, 95
Cargos collection, 984
Cargos property, Annotation class, 983
CellEditEnding event, DataGrid, 748–749
CellSpacing property, 947
CellTemplate property, 712
CellTemplateSelector property, 714
Center property, 357, 377
CenterOwner, 755
CenterX property, 368
CenterY property, 368
certmgr.exe tool, 826
change notification, 110
ChangeVisualState( ) method, FlipPanel control, 571, 578–579
CharacterEllipse option, 961
check boxes, DataGridCheckBoxColumn class, 735–736
CheckAccess( ), DispatcherObject class, 1043
CheckBox class, 179
CheckBox control, 171
CheckBox element, 660, 663
CheckBox.IsChecked property, 128
CheckedListBox control, 1020, 1022
CheckFileExists property, 762
Child property, 185
Children property, 835, 890
chrome classes, 509
Circle value, MarkerStyle property, 944
CircleEase class, 455
Class attribute, 30, 217, 530
class bindings, 282
classes, hierarchy of, 14–17
ClearAllBindings( ) method, 254
ClearBinding( ) method, 254
ClearHighlightsForSelection( ) method, 979
ClearSelection( ) method, 407
ClearValue( ) method, 110, 117, 254
Click event, 129, 177, 844
ClickCount event, 145
ClickMode property, 177
clickOffset field, 406
ClickOnce
accessing installation from UNC path, 1087
adding subdirectories for each new application version, 1092
advantages and disadvantages of, 1079–1080
automatically downloading application updates from Web, 1080
bypassing automatic update feature, 1089
choosing installation type, 1087
choosing publishing location, 1085
choosing update options in Application Updates dialog box, 1095
choosing when application updates are performed, 1096
configuring application for web-like online-only mode, 1082
consumer applications deployed over Web, 1081
contents of.manifest and .application files, 1092
creating application that supports offline use, 1091
creating online-only application, 1091
deployment scenarios and options, 1081
developer control over update settings, 1082
dfsvc.exe, 1094
entering virtual directory on web server for publish location, 1087
enterprise environments and, 1081
inability to install shared components in global assembly cache (GAC), 1082
Install Mode setting, 1094
installation and configuration limitations, 1082
installation model, 1081

link to publish.htm website for application downloading and installation, 1081
location of setup.exe and other deployed files, 1091
making application updates mandatory, 1096
procedure for installing ClickOnce application, 1092–1093
publishing straight to web server using FTP, 1088
publishing to CD or DVD, 1089
publishing to network file share, 1087
publishing to web server or local intranet, 1088
running setup.exe program, 1092
running WPF applications with unmanaged code permission and full trust, 1080
setting miscellaneous options, 1098
setting publish version, 1094
shortcut for ClickOnce applications, 1093
tracking separate assembly and publication version numbers, 1095
transferring application files to web server, 1088
unsuitability for sophisticated consumer applications, 1081
updating ClickOnce application automatically, 1093
using Visual Studio publishing wizard, 1085
using Visual Studio to publish ClickOnce application to web server, 1081
verifying that.NET Framework 3.0 runtime is installed, 1081
ClickOnce cache, 823
client area, definition of, 751
clip art, exporting, 396
Clip property, 391–392
clipboard, 202
ClipGeometry class, 393
Clipping property, 881
ClipToBounds property, 95, 343, 462, 890
Clock class, 452
Clock property, 873
Close( ) method, 755
Closing event, 135
CLR (common language runtime), 13, 242
code and compiled markup (BAML), 26, 48, 53
code and uncompiled markup (XAML), 48, 51
Code DOM model, 54
code only development, 48–49
CodeAccessPermission class, 827
code-based animations, 429
code-behind class, 30
CoerceValueCallback, 111, 115
coercion, 115, 117
CollectionView, 691, 693
CollectionView.GroupDescriptions collection, 703
CollectionViewSource class, 692, 696
color picker
adding basic Grid, 552
adding command bindings, 555
adding command support to controls, 555
adding standard property wrappers, 550
adding support for ApplicationCommands.Undo command, 555
adding TemplatePart attribute to control declaration, 566
adding user control to custom control library project, 548
adjusting color components with Slider controls, 547
attaching callbacks that respond to changed properties, 549
calling OverrideMetadata( ) method, 560
calling RaiseEvent( ) method, 552
changing color picker into lookless control, 560
checking for correct type of element, 564
code for binding SolidColorBrush, 565
Color property, 548, 555
ColorChanged event, 555
complete markup, 552
connecting data binding expression using OnApplyTemplate( ) method, 564
control consumer, 548
converting ordinary markup into control template, 562
creating basic color picker, 547
creating standard .NET event wrapper, 552
creating template for, 562
DefaultStyleKeyProperty, 560
defining and adding routed events, 551

defining static fields for properties, 548–549
definition of, 547
designing public interface, 548
generic.xaml resource dictionary, 560
handling CanExecute and Executed events, 556
mapping assembly and .NET namespace to XML namespace, 554
markup structure for ColorPicker.xaml, 561
not using coercion callbacks with color properties, 551
OnColorChanged( ) method, 552
overriding OnApplyTemplate( )method, 564
property change callbacks for updating properties, 550
providing descriptive names for element names, 563
Red, Green, and Blue properties, 549
revised command handling code, 557
RoutedEventHandler, 552
RoutedPropertyChangedEventHandler, 552
SetValue( ) method, 550
static constructor code for registering dependency properties, 549
streamlining color picker control template, 563
tracking previous color in member field, 555
triggering Undo command, 556
UserControl class, 548
using binding expressions to repurpose core property, 553
using data binding for color sliders, 549
using in another window, 554
using TargetType attribute, 561
Color property, 413, 548, 551
ColorAnimation class, 426, 473, 521
ColorChanged event, 555
ColorDialog class, 1025
ColorDialog control, 1020
Color.FromArgb( ) method, 161
Colors class, 161–162
Column property, 81
ColumnDefinition element, 81, 100
ColumnDefinition object, 83
ColumnGap property, 964, 998
ColumnHeaderContainerStyle property, 715
ColumnHeaderHeight property, DataGrid, 733
ColumnHeaderStyle property, DataGrid, 742
ColumnRuleBrush property, FlowDocument class, 964
ColumnRuleWidth property, FlowDocument class, 964
columns
DataGrid defining, 735
DataGrid formatting and styling, 740–742
DataGridCheckBoxColumn class, 735–736
DataGridColumn class, 735
DataGridTemplateColumn class, 735, 739
DataGridTextColumn class, 735–736
freezing, DataGrid control, 745
IsFrozen property, 745
resizing and rearranging DataGrid columns, 734–735
sorting rows based on columns, DataGrid, 746
wrapping text in, 740
Columns collection, DataGrid, 735
ColumnSpan property, 86, 100, 947
ColumnWidth property, 733, 998
ColumnWidth property, FlowDocument class, 964
CombinedGeometry class, 376
applying transform to geometry, 382
building up distinct shapes with many geometries, 381
combining overlapping shapes, 380
creating simple "No" sign, 381–383
GeometryCombineMode property, enumeration values, 380
merging two shapes to create one, 380
using Geometry1 and Geometry2 properties, 380
ComboBox control
adding complex objects to, 686
AutoComplete feature, 686
ComboBoxItem object, 206
components of, 686
DisplayMemberPath property, 687
hard-coding value for Width property, 207
IsEditable property, 207, 686, 688
IsReadOnly property, 686, 688
IsTextSearchEnabled property, 686
not placing user-interactive controls in drop-down list, 687

SelectionBoxItemTemplate property, 688
setting IsDropDownOpen property, 686
setting TextSearch.TextPath property, 688
use of drop-down list, 207
using automatic sizing, 207
using nontext content in, 686
ComboBoxItem object, 206
Command class, InputBindings collection, 275
Command property, 844
Command, ThumbButtonInfo class, 787
CommandBinding class, 273
CommandBindings collection, 284, 557
CommandHistoryItem class, 287–288
command-line arguments, 223–224
CommandManager class
InvalidateRequerySuggested( ) method, 279, 283
keeping command history, 286
RegisterClassCommandBinding( ) method, 557
CommandParameter property, 273, 286, 844
commands
adding command bindings to top-level window, 273
adding new binding for command you want to disable, 281
adding new command bindings or input bindings to disable features, 282
ApplicationCommands class, 270
basic command library, 270
binding custom command differently in two places, 284
calling static CommandManager.InvalidateRequerySuggested( ) method, 279
calling Undo( ) method of CommandHistoryItem, 291
CanExecute( ) method, 268
CanExecuteChanged event, 268
class bindings, 282
command binding, definition of, 267
command sources, 267–268, 271
command target, definition of, 267
CommandBinding class, 273
CommandHistoryItem class, 287–288
commands as static objects global to application, 271
ComponentCommands class, 270
controls raising CanExecuteChanged event, 279
controls with built-in commands, 280
creating all-purpose, application-wide Undo( ) command, 288–289
creating command binding, 272
custom commands, 282
dealing with user-interface state, 266
default input bindings and command objects, 271
definition of, 267–268, 760
disabling, 272, 277
disabling input bindings, 281
EditingCommands class, 270
Execute( ), 268
features of, 265
forcing WPF to call CanExecute( ) on all used commands, 279
handling commands that vary between enabled and disabled state, 277
ICommand interface, 267
ICommandSource interface, table of properties, 271
instantiating new RoutedUICommand object, 282
localization and setting Text property, 275
mapping events to same command, 266
mapping .NET namespace to XML namespace, 283
MediaCommands class, 270
modifying RoutedCommand.InputGestures collection of command, 283
namespace requirement for using custom command in XAML, 283
NavigationCommands class, 270
not adding unwanted commands to Undo history, 290
pulling text out of static command object, 276
reacting to executed commands using CommandManager, 289
refining Undo feature for real-world application, 291
Requery command, 283
RequerySuggested event, 279
responding to PreviewExecuted event, 289

and ribbons, 855–856
RoutedCommand class, 268
RoutedUICommand class, 268–269
setting Boolean isDirty flag, 278
setting CommandParameter property, 273, 286
setting text for menu access keys, 275
storing CommandHistoryItem objects in ListBox, 290
supplying keyboard shortcut for InputGestures collection, 282
supporting application-wide Undo feature, 286
switching off control's built-in command support, 281
techniques for reusing command text, 276
tracking and reversing commands, 286
using ApplicationCommands.NotACommand value, 281
using command parameter to pass extra information, 286
using CommandManager class to keep command history, 286
using data binding expression to pull out Text property, 276
using event handlers to call appropriate application methods, 265
using Execute( ) to invoke command directly, 276
using multiple command sources, 275
using same command in different places, 284
using WPF resources, 285
when user-interface state falls outside WPF command model, 279
why WPF commands require routed events, 268
wiring up commands declaratively using XAML, 274
WPF command model, 266–267
WPF's prebuilt commands, 268
CommandTarget property, 280, 844
common language runtime (CLR), 13, 242
CommonStates group, 541
Completed event, Storyboard class, 487, 496
component class, 1026
component tray, 1026
ComponentCommands class, 270
ComponentResourceKey, 305–306, 718
composable applications, 1056
Composite Application Library (CAL), 1056
composition, 590
CompositionTarget.Rendering event, 483
ComVisible attribute, 838
Connect( ) method, 53
container controls, 41
ContainerFromElement( ) method, 206
containers, 61, 64
ContainerStyle property, GroupStyle class, 704
ContainerStyleSelector property, GroupStyle class, 704
ContainerUIElement3D, 929
ContainerVisual class, 400, 1002
containment, 27
content controls
button controls and access keys, 177
ButtonBase class, 177
CheckBox class, 179
CheckBox control, 178
Escape key, 177
GridViewColumnHeader class, 178
headered, 159
Label control, 175
mnemonics, 175
overview, 159
Popup control, 185
RadioButton control, 178–179
RepeatButton class, 178
ToggleButton class, 178–179
ToolTip class, 180
ToolTip property, 180
Tooltips control, 180
content elements, 939
Content property, 41, 171, 669, 812
Content property, Page class, 795
ContentBox property, 997
ContentControl class, 16, 42
aligning content relative to its borders, 173
Button control, 171
CheckBox control, 171
class hierarchy of, 170

combining text and images in StackPanel, 172
content controls and content nesting, 175
content controls, definition of, 169
Content property, 545
Content property and UIElement class, 171
ContentTemplate, 173
ContentTemplate property, 545
description of, 545
differentiating content controls from layout containers, 169
displaying text string on button surface, 171
drawing vector image inside button, 174
HasContent property, 173
HeaderedContentControl class, 171
Label, 171
OnRender( ) method, 171
placing image inside button, 171
RadioButton control, 171
ScrollViewer, 171, 188, 192–193
System.Windows.Shapes namespace, 174
ToolTip control, 171
ToString( ) method, 171, 173
UserControl class, 171
using Image class, 171
Window class, 170–171, 751
WPF windows and, 224
ContentControl element, FlipPanel control, 573
ContentElement class, 121, 429, 937
ContentEnd property, 968
ContentLocator object, 982
ContentPresenter, 506, 558–559, 662, 723
required for all content controls, 514
setting RecognizesAccessKey property to true, 515
ContentProperty attribute, 39–41
ContentRendered event, 135
ContentStart property, 968
ContentTemplate, 173, 669
ContentType property, 236
context, 1041
ContextMenu class, 845–846
ContextMenuStrip class, 1026
contract, 1057
contract assemblies, 1063–1064
ContractBase attribute, 1065
ContractHandle object, 1067
Contracts subdirectory, 1064
ContractToViewAdapter( ) method, 1078
Control class
automatic change notification in brushes, 162
Background and Foreground properties, 160
BorderBrush property, 162
BorderThickness property, 162
Brush object, 160
brushes, 160
Cursor property, 168
Cursors class, 168
description of, 545
font embedding, procedure for, 166
font family, definition of, 164
font inheritance, 165
font properties as dependency properties, 165
font substitution, 165
font-related properties of, 163
FontStretches class, 163
FontStyles class, 163
FontWeights class, 163
ForceCursor property, 169
functions of, 160
HorizontalContentAlignment property, 173
identifying FontFamily, 164
ImageBrush, 371
IsTabStop property, 545
LinearGradientBrush, 162
making element partly transparent, 370–372
mouse cursors, 168
MouseDoubleClick event, 145, 545
OnApplyTemplate( ) method, 578
OpenType, 163
OverrideCursor property, 169
padding content of button, 173
Padding property, 173
painting border around controls, 162
PreviewMouseDoubleClick event, 145, 545
property value inheritance, 165
scRGB standard, 161
setting background and foreground colors in XAML, 162
setting button's surface color, 160

setting FontFamily to list of font options, 166
SolidColorBrush, 161
template support, 16
TextDecorations class, 164
TextDecorations property, 164
TileBrush, 371
Typography property, 164
using custom cursor, 169
VerticalContentAlignment property, 173
WPF Color structure, 162
WPF font size compared to Windows point size, 163
control consumer, 548
control templates
adding basic window behaviors to window template, 773
adding control template for ListBoxItem, 531
adding trigger to change button's background, 519
adding triggers to, 517
applying custom control template by setting Template property, 514
applying templates automatically, 527
applying window template using simple style, 772
basic markup for Button control template, 514
basic structural markup for window's control template, 771–772
Border class, 514
browsing WPF control templates, 510
building complex, multipart templates, 530
Button class example, 506
ButtonChrome class, 506
chrome classes, 509
comparing template bindings to data bindings, 516
comparing to custom controls, 526
ContentPresenter, 506
converting live ControlTemplate object to XAML markup, 512
creating, 513
creating code-behind class for resource dictionary, 529, 773
creating focus indicator for button, 518
creating new styles that use same template, 526
creating separate resource dictionary for each control template, 522
creating template for revamped ListBox control, 531
custom controls and user interface standardization, 513
custom controls using different control template, 581–582
customizing template for vertical ScrollBar, 533–536
deciding where to apply templates, 521
defining, 27, 514
defining resources in separate resource dictionaries, 521
defining state animations, 575
defining template details as separate resources, 522
dependencies, 530–531
determining required elements, 578
dissecting controls, 510
IsMouseOver property, 517
IsPressed property, 517
jazzing up customized controls, 541
keeping all triggers in control template, 526
loading resource dictionary defined in another assembly, 529
making window draggable, 773
MergedDictionaries collection, 522, 528
obtaining XAML for standard control templates, 510
organizing template resources, 521
problems in giving new template to common control, 538
providing user-selectable skins, 528
putting template details into associated style, 524
reasons to avoid template bindings, 517
Rectangle class, 514
refactoring Button control template, 522
reflection, 511
RepeatButton class, 534
replacing current resource dictionary at runtime, 528
ResizeGrip element, markup example, 509
resource dictionary for button, complete markup, 522

retrieving control's template and serializing it to XAML, 510
retrieving Padding property using template binding, 515
ScrollBar class, 536
Setter.TargetName property, 526
setting key name on style, 536
setting sequential order for conflicting trigger settings, 519
setting TargetName property of each Setter, 518
similarity between templates and styles, 519
SimpleStyles project, 538
starting default control template, 573–574
styling ScrollBar's RepeatButton objects and Thumb, 537–538
template bindings, 515
Track class, 534
using adorner layer to draw superimposed content, 771
using DynamicResource reference, 529
using focus and click triggers, 507
using FrameworkElement.TemplatedParent property, 773
using resource dictionary, 522
using ResourceManager class, 528
using StaticResource reference, 514
using styles and control templates to skin any application, 527
using TargetName property, 507
using template bindings to pull details out of control properties, 524
when to switch from custom control templates to custom controls, 526
XamlReader class, 513
XamlWriter class, 510
XamlWriter.Save( ) method, 512
ControllableStoryboardAction class
action classes for controlling storyboard, 445
controlling animation playback, 445
defining all triggers in one Triggers collection, 446
storyboard actions and properties, 448
controls
arranging based on their content, 5
background and foreground in, 160
base classes for creating custom element, 544
choosing Custom Control Library (WPF) project type, 544
choosing right base class to inherit from, 544
content controls, definition of, 169
control templates, 506
creating custom control, 544
creating templates for custom controls, 582
creating undo stack that stores series of values, 557
definition of, 16, 160
FlipPanel control, 582
Label, 160
logical tree, building, 500
lookless, 506, 543
mnemonics, 160
never using custom drawing in, 595
past problems in control customization, 499
placing custom controls in dedicated class library (DLL) assembly, 544
ribbon, 859, 862
ToolTip, 160
visual tree, definition of, 501
writeable control properties as usually dependency properties, 548
ControlTemplate, 715
ConvertToString( ) method, 139
Copy Local property, 1066
Copy to Output Directory, 239
CopyPixels( ) method, 417
CornerHeaderStyle property, DataGrid, 742
CornerRadius property, 74, 571
Count property, 692
CreateHighlightCommand, 981
CreateHighlightsForSelection( ) method, 979
CreateInkStickyNoteCommand, 980
CreateInkStickyNoteForSelection( ) method, 979
CreateTextStickyNoteForSelection( ) method, 979
CreateXpsDocumentWriter ( ) method, 1017
CreationTime property, Annotation class, 983
csc.exe compiler, 55
.csproj file, 820
CubicEase class, 455
CultureInfo class, 246

Currency data type, 646
Current property, 224
CurrentChanged event, 693
CurrentDispatcher property, 1042
CurrentGlobalSpeedInvalidated event, 451
CurrentProgress property, 452
CurrentStateInvalidated event, 451
CurrentTimeInvalidated event, 451
CurrentUICulture property, 240
Cursor property, 168
Cursors class, 168
Custom Control Library (WPF), 544, 803
custom controls
  control classes to derive from, 569
  creating templates for
    defining state animations, 575
    FlipPanel control, 582
    starting default control template, 573–574
  exploring alternatives to, 543
  reusing user-interface functionality, 325
  using different control template, 581–582
Custom DPI Setting dialog box, 11
custom panels
  adding attached LineBreakBeforeProperty to WrapBreakPanel, 587
  Arrange( ) as triggering ArrangeOverride( ), 585
  ArrangeCore( ) method, 583
  ArrangeOverride( ) method, 583, 585–586
  basing WrapBreakPanel on WrapPanel, 588
  Canvas class, 586
  creating Canvas-style panel, 586
  examples of, 583
  extending capabilities of WrapPanel, 587
  layout pass, 583
  Measure( ) as triggering MeasureOverride( ), 584
  measure pass, 583
  MeasureCore( ) method, 583
  MeasureOverride( ) method, 583, 586
  RegisterAttached( ) method, 587
  two-step layout process, 583
  uses for, 582
CustomBehaviorsLibrary assembly, 329
CustomCategory property, JumpPath class, 781
CustomContentState class, 811, 815
CustomDrawnDecorator, 596
CustomDrawnElement, 593–594
CustomFilter property, 701
CustomPopupPlacementCallback property, 182–183
CustomSort property, 702

D

dashed lines, 349–351
data binding, 1032
  adding validation rule to Binding.ValidationRules collection, 627
  AdornedElementPlaceholder, 630, 632
  AncestorType property, 262
  automatic target updating, 251
  Binding class, 251
  binding elements closely to their data, 256
  Binding markup extension, 251, 254
  binding to ADO.NET data objects, 615
  binding to collection of objects, 609
  binding to LINQ expression, 617–618
  binding to non-element objects, 260
  binding to nonexistent property, 251
  binding updates, 259
  BindingExpression class, 260
  BindingExpression.UpdateSource( ) method, 259–260
  BindingMode enumeration, table of values, 252
  BindingOperations class, 254
  Binding.RelativeSource property, 681, 683
  bubbling, 628
  building data access components, 600
  building data object, 603
  building validation directly into controls, 621
  caller inform design pattern, 603
  chaining bindings, 256
  change notification and dependency properties, 249
  checking InnerException property of TargetInvocationException, 629
  ClearAllBindings( ) method, 254
  ClearBinding( ) method, 254
  ClearValue( ) method, 254

contents of ValidationError object, 628
creating and using value converter class, 647
creating Binding object with nested RelativeSource object inside, 261
creating binding using code, 254
creating custom controls, 255
creating DataView, 615
creating dynamic binding, 255
creating error templates, 630
creating multiple binding expressions that set same property, 256
creating multiple bindings, 255
creating XAML pages to run in browser, 250
data binding
  collection items, displaying and editing, 610
  collection items, inserting and removing, 614
data conversion, definition of, 645
data templates and, 672
DataContent property, 613
DataContext property, 261, 263, 605
DataErrorValidationRule, 624–625
DataTable.DefaultView property, 615
defining validation at binding level, 621
definition of, 249, 599
displaying bound object, 604
displaying error content in ToolTip, 631
DisplayMemberPath property, 612, 616, 644
element-to-element binding, 249
enabling database updates, 606
ErrorContent property, 627–628
ExceptionValidationRule, 623, 627
Explicit update mode, 260
FindAncestor mode, 262
forcing values to flow bidirectionally between source and target, 252
FormHasErrors( ) method, 630
getting list of all outstanding errors, 629
guidelines for designing data access components, 600
handling change notification, 607
handling Error event, 628
HasError property, 623
how WPF handles validation failures, 623
IEnumerable interface, 610
INotifyCollectionChanged interface, 614
INotifyPropertyChanged interface, 608
IsValid property, 627
ItemsControl class, 643–644, 669
ItemsControl class, table of properties, 609
ItemsSource property, 644, 669, 672
linking controls through, 250
LostFocus update mode, 259
markup-based vs. programmatic binding, 255
modifying data binding source programmatically, 251
NotifyOnValidationError property, 623
OnPropertyChanged( ), 609
options for catching invalid values, 621
outputting trace information on binding failures, 251
preventing field from being edited, 613
PropertyChanged event, 608
PropertyChanged update mode, 259
raising errors in data object, 621–622
reacting to validation errors, 628
reducing overhead by setting mode to one-way binding, 254
RelativeSource property, 260–261
RelativeSourceMode enumeration, table of values, 262
removing binding with code, 254–255
setting DataContext property of container, 644
setting ElementName property, 251
setting Mode property of Binding, 252
setting NotifyOnValidationError property, 628
setting Path property, 251
setting property that isn't dependency property, 254
Source property, 260–261
StoreDB class, 601
summary of data-binding procedure, 644
TargetInvocationException, 629
TemplateBinding, 562–563
two-way bindings, 258, 599
understanding OneWayToSource BindingMode, 253
UpdateSourceTrigger property, 259, 627

using same validation rule for more than one binding, 627
Validation class, 623
validation rule for restricting decimal values, 626
ValidationError object, 623
Validation.ErrorTemplate property, 624
ValidationResult object, 627
ValidationRules collection, 623
ValidationRule.Validate( ) method, 623
visual indication of errors in bound controls, 624
writing custom validation rules, 626
writing data binding expressions, 26
writing data binding expressions in XAML, 251
data conversion
applying conditional formatting, 653
BitmapImage class, 652
converting from display format back to number, 649
converting raw binary data into WPF BitmapImage object, 651
creating converter object in Resources collection, 650
creating objects with value converter, 650
creating value converter class, 647
data triggers, 654
Decimal.ToString( ) method, 649
definition of, 645
evaluating multiple properties, 654
formatting strings with data converter, 648
ImagePathConverter, code example, 651
IMultiValueConveter interface, 655
mapping project namespace to XML namespace prefix, 650
Parse( ) method, 649
PriceConverter class, 650
SuppressExceptions property, 652
System.Globalization.NumberStyles value, 649
TryParse( ) method, 649
using custom IValueConverter, 653
Data property, 375
data providers
defining data object as resource of window or container, 636
functions of, 636
instantiating data objects in XAML, 637
ObjectDataProvider, 637–640
overview, 636–637
System.Windows.Data.DataSourceProvider class, 637
XmlDataProvider, 637, 640
data source, Visual Studio, 618
data templates
adding elements inside existing control, 508
binding Visibility property to IsSelected property, 683
building specialized class deriving from DataTemplateSelector, 665, 676
changing item layout by setting ItemsPanelTemplate property, 684
code examples, 669
composition of, 669
Content property, 669
ContentTemplate property, 669
creating template selector, 676
creating template that adjusts to bound object, 676
data binding and, 672
data triggers, 675
DataGridTemplateColumn class, 735, 739
defining in resources collection, 671
definition of, 669
functions of, 508
ItemTemplate property, 669
list-based and content-control templates, 669
modifying template of selected or deselected item, 679
placing controls directly inside template, 673
presenting different data items in different ways, 675
retrieving all information about selected data item, 674
reusing same data template in different types of controls, 671
SelectTemplate( ) method, 665–666
setting DataType property, 671
setting SnapsToDevicePixels property, 682
setting style of inner StackPanel using trigger, 683

setting template's element property based on data-item property, 675
SingleCriteriaHighlightTemplateSelector class, 667, 677
template selection and displaying editable data, 668
template selectors, 675
using binding expression to alter template, 680
using IValueConverter objects in data binding, 672
using with StaticResource reference, 671
value converters, 675
data triggers, 654, 675–676
data types
data binding format string, 646
DataTypeAnimation class, 479
having or not having corresponding animation class, 426
key frame vs. interpolation animation classes, 426
linear and discrete key frames, 479
setting properties using resource, 310
data views
adding group header, 705
adding multiple levels of grouping, 705
adjusting filtering through WPF view object, 701
ADO.NET DataView, function of, 701
applying grouping, 703
binding ObservableCollection, 691
binding same data in different ways within window, 691
clearing existing SortDescriptions collection, 702
code for connecting IComparer to view, 703
combining grouping with sorting, 705
constructing CollectionViewSource declaratively in XAML, 695
Count property, 692
creating filtering class, 699
creating more than one DataView to wrap same DataTable, 702
creating multiple views, 697
creating Predicate object, 698
creating separate GroupItem object for each group, 703
creating single filtering class for each window, 700
creating views declaratively, 695
CurrentChanged event, 693
defining inline filtering method, 698
definition of, 691
filtering collections, 697
filtering DataTable, 701
filtering dynamically, 698
forcing list to be refiltered, 700
GetDefaultView( ) method, 692
grouping data objects in ranges, 705
ICollectionView interface, 693
implementing IBindingList, 691
implementing IEnumerable, 691
implementing IList, 691
navigating data objects and records with view, 692
removing filter, 700
retrieving bound DataView and modifying its properties directly, 701
retrieving view object, 692
setting GroupStyle.HeaderTemplate property, 705
sorting based on property values in each data item, 702
storing reference to filter object as member variable, 700
using Filter property of view object, 697
using lookup list for editing, 695
using SortDescription to identify sort field and sort direction, 702
using value converter to apply range grouping, 707
using view to implement sorting, 702
view objects and CollectionView, 691
writing logic for previous and next buttons, 694
DataContent property, 613
DataContext property, 261, 263, 605, 644
DataFormats class, 968–969
DataGrid control
AlternatingRowBackground property, 733
AutoGenerateColumns property, 732, 735
BeginningEdit event, 748
CanUserReorderColumns property, 735
CanUserResize property, 735