

1

XML Paper 2 Specification

3 XPS Specification and Reference Guide

4
5
6
7 Working Draft 1.2

8
9 May 2008

10 This document is "a work in progress" and was produced by Ecma Technical Committee TC46.
11
12
13
14

1	Contents	
2	1. SCOPE	1
3	2. CONFORMANCE	3
4	2.1 Language Notes	3
5	2.2 Software Conformance	3
6	3. NORMATIVE REFERENCES	5
7	4. DEFINITIONS	7
8	5. NOTATIONAL CONVENTIONS	11
9	5.1 Document Conventions	11
10	5.2 Diagrams	11
11	6. ACRONYMS AND ABBREVIATIONS	13
12	7. GENERAL DESCRIPTION	15
13	8. XPS DOCUMENT FORMAT	17
14	8.1 How This Specification Is Organized	17
15	8.2 Package	19
16	9. PARTS AND RELATIONSHIPS	21
17	9.1 Fixed Payload	21
18	9.1.1 Fixed Payload Relationships.....	24
19	9.1.2 FixedDocumentSequence Part.....	25
20	9.1.3 FixedDocument Part.....	25
21	9.1.4 FixedPage Part.....	26
22	9.1.5 Image Parts.....	26
23	9.1.6 Thumbnail Parts	32
24	9.1.7 Font Parts	33
25	9.1.8 Remote Resource Dictionary Parts	37
26	9.1.9 PrintTicket Parts.....	37
27	9.1.10 SignatureDefinitions Part	39
28	9.1.11 DocumentStructure Part	40
29	9.1.12 StoryFragments Part	40
30	9.2 Part Naming Recommendations	41
31	9.3 XPS Document Markup	43
32	9.3.1 Support for Versioning and Extensibility	44
33	9.3.2 XML Usage	44
34	9.3.3 Markup Model	45
35	9.3.4 Whitespace.....	47
36	9.3.5 Language	47
37	10. DOCUMENTS	49
38	10.1 <FixedDocumentSequence> Element	49
39	10.1.1 <DocumentReference> Element.....	49
40	10.2 <FixedDocument> Element	50
41	10.2.1 <PageContent> Element.....	50
42	10.2.2 <PageContent.LinkTargets> Element	51
43	10.2.3 <LinkTarget> Element.....	52
44	10.3 <FixedPage> Element	53
45	10.3.1 BleedBox Attribute.....	54

1 10.3.2 ContentBox Attribute 55

2 10.3.3 Page Size Terminology 55

3 10.3.4 Media Orientation and Scaling 56

4 **10.4 <Canvas> Element 61**

5 **10.5 <Path> Element 64**

6 **10.6 <Glyphs> Element 65**

7 **11. GRAPHICS 67**

8 **11.1 <Path> Element 68**

9 11.1.1 <Path.Data> Element 72

10 11.1.2 <Path.Fill> Element 73

11 11.1.3 <Path.Stroke> Element 74

12 **11.2 Geometries and Figures 75**

13 11.2.1 Geometries 77

14 11.2.2 Figures 80

15 11.2.3 Abbreviated Geometry Syntax 89

16 **12. TEXT 97**

17 **12.1 <Glyphs> Element 98**

18 12.1.1 Glyph Metrics 103

19 12.1.2 Mapping Code Units to Glyphs 104

20 12.1.3 Indices Attribute 109

21 12.1.4 UnicodeString Attribute 111

22 12.1.5 StyleSimulations Attribute 111

23 12.1.6 IsSideways Attribute 112

24 12.1.7 DeviceFontName Attribute 118

25 12.1.8 xml:lang Attribute 119

26 12.1.9 CaretStops Attribute 119

27 12.1.10 Optimizing Glyph Markup 119

28 12.1.11 Glyph Markup Examples 121

29 **12.2 <Glyphs.Fill> Element 124**

30 **13. BRUSHES 125**

31 **13.1 <SolidColorBrush> Element 126**

32 **13.2 <ImageBrush> Element 127**

33 **13.3 <VisualBrush> Element 131**

34 13.3.1 <VisualBrush.Visual> Element 132

35 **13.4 Common Attributes for Tiling Brushes 136**

36 13.4.1 Viewbox, Viewport, ViewboxUnits, and ViewportUnits Attributes 136

37 13.4.2 TileMode Attribute 141

38 **13.5 <LinearGradientBrush> Element 152**

39 13.5.1 SpreadMethod Attribute 154

40 13.5.2 <LinearGradientBrush.GradientStops> Element 157

41 **13.6 <RadialGradientBrush> Element 158**

42 13.6.1 SpreadMethod Attribute 162

43 13.6.2 <RadialGradientBrush.GradientStops> Element 165

44 **13.7 <GradientStop> Element 166**

45 **13.8 Using a Brush as an Opacity Mask 167**

46 **14. COMMON PROPERTIES 171**

47 **14.1 Opacity 172**

48 **14.2 Resources and Resource References 172**

49 14.2.1 <FixedPage.Resources> Element 173

50 14.2.2 <Canvas.Resources> Element 175

1	14.2.3	<ResourceDictionary> Element	176
2	14.2.4	Resource References	180
3	14.2.5	Scoping Rules for Resolving Resource References	180
4	14.2.6	Support for Markup Compatibility	182
5	14.3	Clipping	183
6	14.3.1	<Canvas.Clip> Element	183
7	14.3.2	<Path.Clip> Element	184
8	14.3.3	<Glyphs.Clip> Element	185
9	14.4	Positioning Content	186
10	14.4.1	<MatrixTransform> Element	186
11	14.4.2	<Canvas.RenderTransform> Element	190
12	14.4.3	<Path.RenderTransform> Element	191
13	14.4.4	<Glyphs.RenderTransform> Element	192
14	14.4.5	<PathGeometry.Transform> Element	193
15	14.4.6	<ImageBrush.Transform> Element	195
16	14.4.7	<VisualBrush.Transform> Element	196
17	14.4.8	<LinearGradientBrush.Transform> Element	199
18	14.4.9	<RadialGradientBrush.Transform> Element	200
19	14.5	OpacityMask	203
20	14.5.1	<Canvas.OpacityMask> Element	203
21	14.5.2	<Path.OpacityMask> Element	204
22	14.5.3	<Glyphs.OpacityMask> Element	205
23	15. COLOR		209
24	15.1	Color Support	209
25	15.1.1	sRGB Color Space	209
26	15.1.2	scRGB Color Space	210
27	15.1.3	Gray Color Space	210
28	15.1.4	CMYK Color Space	210
29	15.1.5	N-Channel Color Spaces	210
30	15.1.6	Named Color for Spot Colors and N-tone Images	210
31	15.1.7	Device Color Spaces	210
32	15.1.8	ICC Profiles	210
33	15.1.9	WcsProfilesTag	211
34	15.1.10	WCS Color Profiles	212
35	15.2	Vector Color Syntax	212
36	15.2.1	sRGB Color Syntax	213
37	15.2.2	scRGB Color Syntax	214
38	15.2.3	CMYK Color Syntax	215
39	15.2.4	N-Channel Color Syntax	215
40	15.2.5	Named Color Syntax	216
41	15.3	Colors in Raster Images	217
42	15.3.1	sRGB Raster Images	217
43	15.3.2	scRGB Raster Images	218
44	15.3.3	Gray Raster Images	218
45	15.3.4	CMYK Raster Images	218
46	15.3.5	N-channel Raster Images	219
47	15.3.6	Named Color Raster Images	219
48	15.3.7	Device Color Raster Images	220
49	15.3.8	Images and Color Profile Association	220
50	15.3.9	Color Space Pixel Formats for Raster Images	220
51	15.4	Color Separation	221
52	15.5	Alpha and Gradient Blending	221
53	15.6	PrintTicket Color Settings	222
54	16. DOCUMENT STRUCTURE AND INTERACTIVITY		227
55	16.1	Document Structure Markup	227
56	16.1.1	DocumentStructure Part	227
57	16.1.2	StoryFragments Part	233

1 **16.2 Hyperlinks.....247**

2 16.2.1 Hyperlink Activation.....247

3 16.2.2 Hyperlink Addressing248

4 16.2.3 Name Attribute248

5 16.2.4 FixedPage.NavigateUri Attribute.....249

6 **16.3 Selection.....250**

7 **16.4 Accessibility.....250**

8 16.4.1 Reading Order250

9 16.4.2 Screen Reader Applications251

10 16.4.3 Text Alternatives for Graphics and Images.....251

11 **17. XPS DOCUMENT PACKAGE FEATURES.....253**

12 **17.1 Interleaving Optimizations253**

13 17.1.1 Empty PrintTicket255

14 17.1.2 Optimizing Interleaving Order.....255

15 17.1.3 Consuming Interleaved Packages258

16 17.1.4 Consumers with Resource Constraints.....258

17 17.1.5 Interleaving Optimizations and Digital Signatures260

18 **17.2 Digital Signatures.....260**

19 17.2.1 Signature Policy261

20 17.2.2 Signature Definitions.....264

21 **17.3 Core Properties268**

22 **18. RENDERING RULES.....269**

23 **18.1 Coordinate System and Rendering Placement.....269**

24 18.1.1 Page Dimensions.....269

25 18.1.2 Rounding of Coordinates.....269

26 18.1.3 Transforms.....270

27 18.1.4 Pixel Center Location, Pixel Placement, and Pixel Inclusion270

28 18.1.5 Maximum Placement Error271

29 18.1.6 Pixel Placement for Glyphs271

30 18.1.7 Abutment of Shapes271

31 18.1.8 Clipping Behavior271

32 **18.2 Implementation Limits272**

33 **18.3 Gradient Computations273**

34 18.3.1 All Gradients.....273

35 18.3.2 Linear Gradients.....275

36 18.3.3 Radial Gradients.....277

37 **18.4 Opacity Computations.....280**

38 18.4.1 Pre-Multiplied Alpha and Superluminous Colors282

39 **18.5 Composition Rules.....283**

40 18.5.1 Optimization Guidelines284

41 18.5.2 Composition Examples285

42 **18.6 Stroke Rendering.....288**

43 18.6.1 Stroke Edge Parallelization.....288

44 18.6.2 Phase Control288

45 18.6.3 Symmetry of Stroke Drawing Algorithms288

46 18.6.4 Rules for Dash Cap Rendering.....289

47 18.6.5 Rules for Line Cap Rendering.....291

48 18.6.6 Line Caps for Dashed Strokes292

49 18.6.7 Rules for Line Join Rendering.....293

50 18.6.8 Rules for Degenerate Line and Curve Segments.....297

51 18.6.9 Stroking and Fill Rule298

52 18.6.10 Mixing Stroked and Non-Stroked Segments298

53 18.6.11 Stroke Behavior with Multiple Path Figures298

54 18.6.12 Consistent Nominal Stroke Width298

55 **18.7 Brushes and Images299**

56 18.7.1 Small Tiles299

1	18.7.2	Image Scaling	299
2	18.7.3	Tile Placement.....	299
3	18.7.4	Tiling Transparent Visual Brushes and Image Brushes.....	300
4	19.	ELEMENTS.....	301
5	19.1	ArcSegment.....	301
6	19.2	Canvas	302
7	19.3	Canvas.Clip.....	304
8	19.4	Canvas.OpacityMask	304
9	19.5	Canvas.RenderTransform	305
10	19.6	Canvas.Resources	305
11	19.7	Discard.....	305
12	19.8	DiscardControl.....	306
13	19.9	DocumentOutline.....	306
14	19.10	DocumentReference.....	306
15	19.11	DocumentStructure.....	307
16	19.12	DocumentStructure.Outline.....	307
17	19.13	FigureStructure	308
18	19.14	FixedDocument.....	308
19	19.15	FixedDocumentSequence	308
20	19.16	FixedPage	308
21	19.17	FixedPage.Resources	310
22	19.18	Glyphs.....	310
23	19.19	Glyphs.Clip	315
24	19.20	Glyphs.Fill	315
25	19.21	Glyphs.OpacityMask.....	315
26	19.22	Glyphs.RenderTransform.....	316
27	19.23	GradientStop	316
28	19.24	ImageBrush.....	317
29	19.25	ImageBrush.Transform	318
30	19.26	Intent	319
31	19.27	LinearGradientBrush	319
32	19.28	LinearGradientBrush.GradientStops	321
33	19.29	LinearGradientBrush.Transform	321
34	19.30	LinkTarget.....	321
35	19.31	ListItemStructure	322
36	19.32	ListStructure	322
37	19.33	MatrixTransform.....	323
38	19.34	NamedElement	323
39	19.35	OutlineEntry	324
40	19.36	PageContent.....	324
41	19.37	PageContent.LinkTargets	325

1 **19.38 ParagraphStructure** 325

2 **19.39 Path** 326

3 **19.40 Path.Clip** 331

4 **19.41 Path.Data** 331

5 **19.42 Path.Fill** 331

6 **19.43 Path.OpacityMask** 331

7 **19.44 Path.RenderTransform** 332

8 **19.45 Path.Stroke** 332

9 **19.46 PathFigure** 333

10 **19.47 PathGeometry** 333

11 **19.48 PathGeometry.Transform** 334

12 **19.49 PolyBezierSegment** 335

13 **19.50 PolyLineSegment** 335

14 **19.51 PolyQuadraticBezierSegment** 336

15 **19.52 RadialGradientBrush** 336

16 **19.53 RadialGradientBrush.GradientStops** 339

17 **19.54 RadialGradientBrush.Transform** 339

18 **19.55 ResourceDictionary** 339

19 **19.56 SectionStructure** 340

20 **19.57 SignBy** 341

21 **19.58 SignatureDefinition** 341

22 **19.59 SignatureDefinitions** 342

23 **19.60 SigningLocation** 342

24 **19.61 SolidColorBrush** 342

25 **19.62 SpotLocation** 343

26 **19.63 Story** 343

27 **19.64 StoryBreak** 344

28 **19.65 StoryFragment** 344

29 **19.66 StoryFragments** 345

30 **19.67 StoryFragmentReference** 346

31 **19.68 TableCellStructure** 346

32 **19.69 TableRowGroupStructure** 347

33 **19.70 TableRowStructure** 347

34 **19.71 TableStructure** 348

35 **19.72 VisualBrush** 348

36 **19.73 VisualBrush.Transform** 350

37 **19.74 VisualBrush.Visual** 351

1	A. SIGNATURE DEFINITIONS W3C SCHEMA	353
2	B. XPS DOCUMENT W3C SCHEMA	355
3	C. RESOURCE DICTIONARY KEY W3C SCHEMA	379
4	D. DOCUMENT STRUCTURE W3C SCHEMA.....	381
5	E. DISCARD CONTROL W3C SCHEMA	387
6	F. ABBREVIATED GEOMETRY SYNTAX ALGORITHM	389
7	G. scRGB GAMUT BOUNDARY DEFINITION.....	395
8	H. STANDARD NAMESPACES AND CONTENT TYPES.....	401
9	H.1 XML Namespace URIs	401
10	H.2 Content Types	402
11	H.3 Relationship Types	403
12	I. CONFORMANCE REQUIREMENTS	405
13	I.1 XPS Document Format	406
14	I.1.1 MUST Conformance Requirements	406
15	I.2 Parts and Relationships	406
16	I.2.1 MUST Conformance Requirements	406
17	I.2.2 SHOULD Conformance Requirements	413
18	I.2.3 OPTIONAL Conformance Requirements	417
19	I.3 Documents	419
20	I.3.1 MUST Conformance Requirements	419
21	I.3.2 SHOULD Conformance Requirements	420
22	I.4 Graphics.....	421
23	I.4.1 MUST Conformance Requirements	421
24	I.4.2 SHOULD Conformance Requirements	421
25	I.4.3 OPTIONAL Conformance Requirements	421
26	I.5 Text	422
27	I.5.1 MUST Conformance Requirements	422
28	I.5.2 SHOULD Conformance Requirements	424
29	I.5.3 OPTIONAL Conformance Requirements	424
30	I.6 Brushes.....	425
31	I.6.1 MUST Conformance Requirements	425
32	I.7 Common Properties	425
33	I.7.1 MUST Conformance Requirements	425
34	I.7.2 OPTIONAL Conformance Requirements	426
35	I.8 Color	427
36	I.8.1 MUST Conformance Requirements	427
37	I.8.2 SHOULD Conformance Requirements	429
38	I.8.3 OPTIONAL Conformance Requirements	430
39	I.9 Document Structure and Interactivity	431
40	I.9.1 MUST Conformance Requirements	431
41	I.9.2 SHOULD Conformance Requirements	432
42	I.9.3 OPTIONAL Conformance Requirements	434
43	I.10 XPS Document Package Features.....	435
44	I.10.1 MUST Conformance Requirements	435
45	I.10.2 SHOULD Conformance Requirements	437
46	I.10.3 OPTIONAL Conformance Requirements	439
47	I.11 Rendering Rules	440

1 I.11.1 MUST Conformance Requirements440

2 I.11.2 SHOULD Conformance Requirements441

3 I.11.3 OPTIONAL Conformance Requirements444

4 **I.12 Additional Conformance Requirements 446**

5 I.12.1 MUST Conformance Requirements446

6 **J. BIBLIOGRAPHY 447**

7 **K. INDEX 449**

8

1	List of Figures	
2	Figure 8–1. Package-based XPS Document format.....	19
3	Figure 10–1. Page regions.....	56
4	Figure 10–2. Matching PrintTicket and fixed page size and orientation	57
5	Figure 10–3. Matching PrintTicket and fixed page size with differing orientation	57
6	Figure 10–4. Matching PrintTicket and fixed page orientation with differing size	58
7	Figure 11–1. Fill using EvenOdd algorithm.....	79
8	Figure 11–2. Fill using NonZero algorithm	79
9	Figure 11–3. Arc choice A.....	83
10	Figure 11–4. Arc choice B.....	83
11	Figure 11–5. Arc choice C.....	83
12	Figure 11–6. Arc choice D	83
13	Figure 12–1. Glyph metrics	103
14	Figure 12–2. Upright (usually horizontal) glyph metrics.....	103
15	Figure 12–3. Sideways (usually vertical) glyph metrics	104
16	Figure 17–1. A sample signature spot	267
17	Figure 18–1. Extreme curvatures and dash rendering	288
18	Figure 18–2. Flat dash caps.....	289
19	Figure 18–3. Square dash caps	289
20	Figure 18–4. Round dash caps	290
21	Figure 18–5. Triangular dash caps.....	291
22	Figure 18–6. Overlapping dash segments	291
23	Figure 18–7. Flat start line cap, flat end line cap	292
24	Figure 18–8. Square start line cap, square end line cap	292
25	Figure 18–9. Triangular start line cap, triangular end line cap	292
26	Figure 18–10. Round start line cap, round end line cap	292
27	Figure 18–11. Stroke start or end point within a dash for flat dash caps.....	292
28	Figure 18–12. Stroke start or end point within a dash for non-flat dash caps	292
29	Figure 18–13. Stroke start or end point within a gap for flat dash caps	293
30	Figure 18–14. Stroke start or end point within a gap for not-flat dash caps	293
31	Figure 18–15. Round line join with right angle	293
32	Figure 18–16. Round line join with acute angle	294
33	Figure 18–17. Round line join with obtuse angle	294
34	Figure 18–18. Beveled line join with right angle	294
35	Figure 18–19. Beveled line join with acute angle	295
36	Figure 18–20. Beveled line join with obtuse angle	295
37	Figure 18–21. Mitered line join with right angle and miter limit of 1.0	296
38	Figure 18–22. Mitered line join with acute angle and miter limit of 1.0	296
39	Figure 18–23. Mitered line join with obtuse angle and miter limit of 1.0	296
40	Figure 18–24. Mitered line join with right angle and miter limit of 2.0	297
41	Figure 18–25. Mitered line join with acute angle and miter limit of 2.0	297
42	Figure 18–26. Mitered line join with acute angle and miter limit of 10.0	297

43

1 List of Tables

2	Table 9–1. XPS Document parts	21
3	Table 9–2. Fixed payload relationships	24
4	Table 9–3. Supported JPEG APPn markers	26
5	Table 9–4. Support for ancillary PNG chunks	27
6	Table 9–5. Supported TIFF tags	28
7	Table 9–6. Supported Windows Media Photo features	32
8	Table 9–7. Guidelines for OpenType font embedding	34
9	Table 9–8. Cmap table selection	36
10	Table 11–1. Arc segment definition	82
11	Table 11–2. Commands	90
12	Table 12–3. Glyph specifications	109
13	Table 12–4. Portions of the cluster specification	110
14	Table 12–5. IsSideways and BidiLevel effects on origin placement	114
15	Table 13–1. Brush types	125
16	Table 13–2. Common attributes for <ImageBrush> and <VisualBrush> elements	136
17	Table 14–1. Common property attributes	171
18	Table 14–2. Common property elements	172
19	Table 15–1. WcsProfilesTagType structure	211
20	Table 15–2. Syntax summary	213
21	Table 15–3. Color Space Pixel Format Defaults	220
22	Table 15–4. PrintTicket color settings	222
23	Table 16–1. StoryFragments part elements	233
24	Table 16–2. Unicode character categories	249
25	Table 17–1. JobDigitalSignatureProcessing PrintTicket settings	263
26	Table 18–1. Recommended minimum processing requirements	272
27	Table 18–2. Opacity computation symbols	280
28	Table H–1. Package-wide namespaces	401
29	Table H–2. XPS Document namespaces	401
30	Table H–3. Package-wide content types	402
31	Table H–4. XPS Document content types	402
32	Table H–5. Package-wide relationship types	403
33	Table H–6. XPS Document relationship types	403
34	Table I–1. XPS Document format MUST conformance requirements	406
35	Table I–2. Parts and Relationships MUST conformance requirements	406
36	Table I–3. Parts and Relationships SHOULD conformance requirements	413
37	Table I–4. Parts and Relationships OPTIONAL conformance requirements	417
38	Table I–5. Document MUST conformance requirements	419
39	Table I–6. Document SHOULD conformance requirements	420
40	Table I–7. Graphics MUST conformance requirements	421
41	Table I–8. Graphics SHOULD conformance requirements	421
42	Table I–9. Graphics OPTIONAL conformance requirements	421
43	Table I–10. Text MUST conformance requirements	422
44	Table I–11. Text SHOULD conformance requirements	424
45	Table I–12. Text OPTIONAL conformance requirements	424
46	Table I–13. Brushes MUST conformance requirements	425
47	Table I–14. Common properties MUST conformance requirements	425
48	Table I–15. Common properties OPTIONAL conformance requirements	426
49	Table I–16. Color MUST conformance requirements	427
50	Table I–17. Color SHOULD conformance requirements	429
51	Table I–18. Color OPTIONAL conformance requirements	430
52	Table I–19. Document structure MUST conformance requirements	431

1 Table I-20. Document structure SHOULD conformance requirements 432
2 Table I-21. Document structure OPTIONAL conformance requirements 434
3 Table I-22. XPS Document package feature MUST conformance requirements 435
4 Table I-23. XPS Document package feature SHOULD conformance requirements 437
5 Table I-24. XPS Document package feature OPTIONAL conformance requirements 439
6 Table I-25. Rendering rules MUST conformance requirements 440
7 Table I-26. Rendering rules SHOULD conformance requirements 441
8 Table I-27. Rendering rules OPTIONAL conformance requirements 444
9 Table I-28. Additional MUST conformance requirements 446

10

1 List of Examples

2	Example 9-1. A typical XPS Document.....	23
3	Example 9-2. XPS Document part naming.....	42
4	Example 9-3. Property attribute syntax	46
5	Example 9-4. Property element syntax	47
6	Example 10-1. <FixedDocumentSequence> usage.....	49
7	Example 10-2. <FixedDocument> usage	50
8	Example 10-3. <PageContent> usage	51
9	Example 10-4. <PageContent.LinkTargets> usage.....	52
10	Example 10-5. Fixed page markup.....	54
11	Example 10-6. Canvas composition.....	63
12	Example 11-1. <Path.Data> usage	72
13	Example 11-2. <Path.Fill> usage.....	74
14	Example 11-3. <Path.Stroke> usage	75
15	Example 11-4. <PathGeometry> usage.....	78
16	Example 11-5. <ArcSegment> usage.....	83
17	Example 11-6. <PolyBezierSegment> usage	85
18	Example 11-7. <PolyLineSegment> usage.....	86
19	Example 11-8. <PolyQuadraticBezierSegment> usage	88
20	Example 11-9. Closed <PathFigure> usage.....	88
21	Example 11-10. A path described using abbreviated syntax	92
22	Example 11-11. Smooth Bézier curve.....	93
23	Example 11-12. Relative commands and curves	94
24	Example 12-1. One-to-one cluster map	105
25	Example 12-2. Many-to-one cluster map	105
26	Example 12-3. One-to-many cluster map	106
27	Example 12-4. Many-to-many cluster map.....	107
28	Example 12-5. Using indices to specify advance width.....	110
29	Example 12-6. Using the Indices attribute to specify glyph replacement for a cluster	111
30	Example 12-7. Text with positive uOffset and vOffset Indices values.....	115
31	Example 12-8. Right-to-left text (odd BidiLevel)	115
32	Example 12-9. Sideways text (IsSideways set to true)	116
33	Example 12-10. Vertical text.....	116
34	Example 12-11. Japanese vertical text	117
35	Example 12-12. Using the CaretStops attribute to determine a valid caret stop position	119
36	Example 12-13. Basic italic font.....	121
37	Example 12-14. Italic font using StyleSimulations attribute.....	121
38	Example 12-15. Kerning	122
39	Example 12-16. Ligatures	122
40	Example 12-17. Cluster maps	123
41	Example 13-1. <SolidColorBrush> usage.....	127
42	Example 13-2. <ImageBrush> usage.....	129
43	Example 13-3. <VisualBrush.Visual> usage	134
44	Example 13-4. ViewboxUnits and ViewportUnits attribute usage	137
45	Example 13-5. Tiling brush base image and rendering.....	137
46	Example 13-6. Tiling brush Viewport adjustments.....	138
47	Example 13-7. Tiling brush viewbox adjustments.....	139
48	Example 13-8. Image brush with a Viewbox larger than the image	140
49	Example 13-9. Image brush with TileMode value of None	141
50	Example 13-10. Visual brush with TileMode value of None	143
51	Example 13-11. Image brush with a TileMode value of Tile	144
52	Example 13-12. Visual brush with a TileMode value of Tile.....	145

1	Example 13–13. Image brush with a TileMode value of FlipX	146
2	Example 13–14. Visual brush with a TileMode value of FlipX	147
3	Example 13–15. Image brush with a TileMode value of FlipY.....	148
4	Example 13–16. Visual Brush with a TileMode value of FlipY	149
5	Example 13–17. Image brush with a TileMode value of FlipXY.....	150
6	Example 13–18. Visual brush with a TileMode value of FlipXY	151
7	Example 13–19. <LinearGradientBrush> usage	153
8	Example 13–20. Linear gradient brush with a SpreadMethod value of Pad	154
9	Example 13–21. Linear gradient brush with a SpreadMethod value of Reflect.....	155
10	Example 13–22. Linear gradient brush with a SpreadMethod value of Repeat	156
11	Example 13–23. A radial gradient brush.....	161
12	Example 13–24. RadialGradientBrush usage	161
13	Example 13–25. Radial gradient brush with a SpreadMethod value of Pad	162
14	Example 13–26. Radial gradient brush with a SpreadMethod value of Reflect.....	163
15	Example 13–27. Radial gradient brush with a SpreadMethod value of Repeat	164
16	Example 13–28. Opacity mask with linear gradient.....	167
17	Example 13–29. Opacity mask with radial gradient.....	169
18	Example 14–1. <FixedPage.Resources> usage	173
19	Example 14–2. <Canvas.Resources> usage	175
20	Example 14–3. Resource dictionary markup	177
21	Example 14–4. A remote resource dictionary and reference.....	178
22	Example 14–5. Using a resource reference to fill a brush.....	180
23	Example 14–6. Using scoping rules	181
24	Example 14–7. Canvas clip markup and rendering.....	183
25	Example 14–8. <Path.Clip> usage	184
26	Example 14–9. <Glyphs.Clip> usage	185
27	Example 14–10. Matrix scaling.....	187
28	Example 14–11. Matrix reversing the x axis	187
29	Example 14–12. Matrix reversing the y axis	188
30	Example 14–13. Matrix skewing.....	188
31	Example 14–14. Matrix Rotating	188
32	Example 14–15. Matrix positioning	188
33	Example 14–16. <MatrixTransform> usage	188
34	Example 14–17. Using abbreviated matrix transformation syntax.....	190
35	Example 14–18. <Canvas.RenderTransform> usage.....	190
36	Example 14–19. <Path.RenderTransform> usage.....	191
37	Example 14–20. <Glyphs.RenderTransform> usage	192
38	Example 14–21. <PathGeometry.Transform> usage.....	193
39	Example 14–22. <ImageBrush.Transform> usage.....	195
40	Example 14–23. <VisualBrush.Transform> usage	196
41	Example 14–24. <VisualBrush.Transform> usage with tiling behavior.....	198
42	Example 14–25. <LinearGradientBrush.Transform> usage	199
43	Example 14–26. <RadialGradientBrush.Transform> usage	201
44	Example 14–27. <Canvas.OpacityMask> usage	203
45	Example 14–28. <Path.OpacityMask> usage	204
46	Example 14–29. <Glyphs.OpacityMask> usage	207
47	Example 16–1. Document structure markup	228
48	Example 16–2. Document outline markup	230
49	Example 16–3. Simple multi-story document.....	233
50	Example 16–4. Story flowing back and forth across a page boundary	233
51	Example 16–5. Content structure spanning pages	235
52	Example 16–6. StoryFragments part markup.....	240
53	Example 16–7. Story fragments markup using a fragment name.....	241

1 Example 16–8. A relative, internal, named-address hyperlink 248

2 Example 16–9. A relative internal page address hyperlink 248

3 Example 17–1. Optimized interleaving for a single-threaded parsing architecture 255

4 Example 17–2. Optimized interleaving for a multi-threaded parsing architecture 257

5 Example 17–3. A DiscardControl part..... 259

6 Example 17–4. A SignatureDefinitions part..... 265

7 Example 18–1. Path opacity behavior for overlapping path figures 285

8 Example 18–2. Opacity behavior of path stroke intersections..... 285

9 Example 18–3. Opacity behavior of paths with stroked edges 286

10

1. Scope

2 This specification defines *XPS*, the XML Paper Specification. XPS describes the set of
3 conventions for the use of XML and other widely available technologies to describe the content
4 and appearance of paginated documents. It is written for developers who are building systems
5 that process XPS content.

6 A primary goal is to ensure the interoperability of independently created software and hardware
7 systems that produce or consume XPS content. This specification defines the formal
8 requirements that producers and consumers must satisfy in order to achieve interoperability.

9 This specification describes a paginated-document format called the *XPS Document*. The format
10 requirements are an extension of the packaging requirements described in the Open Packaging
11 Conventions (OPC) specification. That specification describes packaging and physical format
12 conventions for the use of XML, Unicode, ZIP, and other technologies and specifications, to
13 organize the content and resources that make up any document. They are an integral part of
14 the XPS specification, and are included by reference.

15 Many XML-based building blocks within XPS make use of the conventions described in the
16 Markup Compatibility and Extensibility specification that is relied upon by the OPC specification
17 to facilitate future enhancement and extension of XPS markup. As such, that Markup
18 Compatibility and Extensibility specification is included by reference.

2. Conformance

2.1 Language Notes

In this specification, the words that are used to define the significance of each requirement are written in uppercase. These words are used in accordance with their definitions in RFC 2119, and their respective meanings are reproduced below:

- **MUST:** This word, or the adjective "REQUIRED", means that the item is an absolute requirement of the specification.
- **SHOULD:** This word, or the adjective "RECOMMENDED", means that there might exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
- **MAY:** This word, or the adjective "OPTIONAL", means that this item is truly optional.

The words **MUST NOT**, **SHOULD NOT**, and **NOT RECOMMENDED**, are the negative forms of **MUST**, **SHOULD**, and **RECOMMENDED**, respectively. There is no negative form of **MAY**.

2.2 Software Conformance

Most requirements are expressed as format or package requirements rather than implementation requirements.

In order for a consumer to be considered conformant, the following rules apply:

- It **MUST NOT** report errors when processing conforming instances of the documented formats except when forced to do so by resource exhaustion.
- It **SHOULD** report errors when processing non-conforming instances of the documented formats when doing so does not pose an undue processing or performance burden.

In order for a producer to be considered conformant, the following rules apply:

- It **MUST NOT** generate any new, non-conforming instances of a documented format.
- It **MUST NOT** introduce any non-conformance when modifying an instance of a documented format.

An application can be both a consumer and a producer.

Conformance requirements are documented inline in this specification, and each requirement is denoted by a letter (M – MUST; S – SHOULD; O – OPTIONAL) and a rule number of the form m.n, where m and n are positive integers, all enclosed in brackets ([...]). [*Example:* [M1.2] is a MUST requirement, [S2.4] is a SHOULD requirement, and [O3.9] is a MAY requirement. *end example*] For convenient reference, these rules are collected in §I.

3. Normative References

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this specification are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

Adobe Photoshop® TIFF Technical Notes, March 22, 2002.

<http://partners.adobe.com/public/developer/en/tiff/TIFFphotoshop.pdf>

BNF of Generic URI Syntax. World Wide Web Consortium.

http://www.w3.org/Addressing/URL/5_URI_BNF.html

Digital Compression and Coding of Continuous-tone Still Images. International Telecommunication Union (ITU). 1993. <http://www.w3.org/Graphics/JPEG/itu-t81.pdf>

ECMA-376, *Office Open XML File Formats* (December 2006), Part 2, "OPC", which is commonly referred to as OPC.

ECMA-376, *Office Open XML File Formats* (December 2006), Part 5, "Markup Compatibility and Extensibility".

Exchangeable Image File Format for Digital Still Cameras: Exif Version 2.2. Technical Standardization Committee on AV & IT Storage Systems and Equipment. Japan Electronic Industry Development Association. 2002. <http://www.jeita.or.jp>

Extensible Markup Language (XML) 1.0 (Fourth Edition). Bray, Tim, Eve Maler, Jean Paoli, C. M. Sperberg-McQueen, and François Yergeau (editors). World Wide Web Consortium. 2006. <http://www.w3.org/TR/2006/REC-xml-20060816/>

HTML 4.01 Specification. Jacobs, Ian, Arnaud Le Hors, and Dave Raggett (editors). World Wide Web Consortium. 1999. <http://www.w3.org/TR/1999/REC-html401-19991224/>

ICC.1:2001-04 File Format for Color Profiles. International Color Consortium. 2001.

http://www.color.org/ICC_Minor_Revision_for_Web.pdf

~~*ICC Profile Format Specification, Version 3.4.* International Color Consortium. 1997.~~

~~<http://www.color.org/icc34.pdf>~~

IEC 61966:2003, *Multimedia systems and equipment - Colour measurement and management - Part 2-2: Colour management - Extended RGB colour space - scRGB*

ISO 15076-1, Image technology colour management — Architecture, profile format, and data structure — Part 1: Based on ICC.1:2004-10

ISO/IEC 2382.1:1993, *Information technology — Vocabulary — Part 1: Fundamental terms.*

ISO/IEC 10646:2003 (all parts), *Information technology — Universal Multiple-Octet Coded Character Set (UCS).*

- 1 *JPEG File Interchange Format, Version 1.02*. Hamilton, Eric. World Wide Web Consortium. 1992.
2 <http://www.w3.org/Graphics/JPEG/jif3.pdf>
- 3 *Namespaces in XML 1.0 (Second Edition)*. Bray, Tim, Dave Hollander, Andrew Layman, and
4 Richard Tobin (editors). World Wide Web Consortium. 2006. [http://www.w3.org/TR/2006/REC-](http://www.w3.org/TR/2006/REC-xml-names-20060816/)
5 [xml-names-20060816/](http://www.w3.org/TR/2006/REC-xml-names-20060816/)
- 6 *Portable Network Graphics (PNG) Specification*. Duce, David (editor). Second Edition. World
7 Wide Web Consortium. 2003. <http://www.w3.org/TR/2003/REC-PNG-20031110>
- 8 RFC 2045, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message*
9 *Bodies*. Borenstein, N., and N. Freed. The Internet Society. 1996.
10 <http://www.ietf.org/rfc/rfc2045.txt>.
- 11 RFC 2119 — *Key words for use in RFCs to Indicate Requirement Levels*. Bradner, S. The
12 Internet Society. 1997. <http://www.rfc-editor.org>
- 13 RFC 3066 — *Tags for the Identification of Languages*. Alvestrand, H. The Internet Society. 2001.
14 <http://www.rfc-editor.org>
- 15 RFC 4234 — *Augmented BNF for Syntax Specifications: ABNF*. Crocker, D. (editor). The Internet
16 Society. 2005. <http://www.rfc-editor.org>
- 17 *A Standard Default Color Space for the Internet—sRGB, Version 1.10*. Anderson, Matthew,
18 Srinivasan Chandrasekar, Ricardo Motta, and Michael Stokes. World Wide Web Consortium.
19 1996. <http://www.w3.org/Graphics/Color/sRGB>
- 20 *TIFF, Revision 6.0*. Adobe Systems Incorporated. 1992.
21 <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>
- 22 *Unicode Character Database, Revision 4.0.0*. Davis, Mark and Ken Whistler. The Unicode
23 Consortium. 2003. <http://www.unicode.org/Public/4.0-Update/UCD-4.0.0.html>
- 24 *The Unicode Standard, Version 4.0*. The Unicode Consortium. Boston, MA: Addison-Wesley,
25 2003, ISBN 0-321-18578-1.
- 26 *XML Base*. Marsh, Jonathan. World Wide Web Consortium. 2001.
27 <http://www.w3.org/TR/2001/REC-xmlbase-20010627/>
- 28 *XML Schema Part 1: Structures, Second Edition*. Beech, David, Murray Maloney, Noah
29 Mendelsohn, and Henry S. Thompson (editors). World Wide Web Consortium. 2004.
30 <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>
- 31 *XML Schema Part 2: Datatypes, Second Edition*. Biron, Paul V. and Ashok Malhotra (editors).
32 World Wide Web Consortium. 2004. <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>

1 4. Definitions

2 For the purposes of this specification, the following definitions apply. Terms explicitly defined in
3 this specification are not to be presumed to refer implicitly to similar terms defined elsewhere.

4 **alpha blending** — Transparently blending two elements when rendering.

5 **consumer** — A piece of software or a device that reads XPS packages.

6 **content structure** — The set of markup elements that allow expression of well-understood
7 semantic blocks, such as paragraphs, tables, lists, and figures.

8 **content type** — Describes the type of content stored in a part. Content types define a media
9 type, a subtype, and an optional set of parameters, as defined in RFC 2045.

10 **coordinate space, effective** — The default coordinate space (X,Y in the upper-left corner,
11 units of 1/96") as modified by any RenderTransform or Transform attributes of the current
12 element and any ancestor elements.

13 **contour intersection point** — The intersection of the flat line ending a dash and the contour
14 of the shape.

15 **device** — A piece of hardware, such as a printer or scanner, that performs a single function or
16 a set of functions.

17 **digital signature, broken** — A digital signature that conforms to the XPS Document signing
18 rules but does not meet the digital signature validity requirements due to incorrect hash
19 calculation or similar problems.

20 **digital signature, compliant** — A digital signature that conforms to the signing rules
21 described in the XPS Document signing policy, regardless of signature validity.

22 **digital signature, non-compliant** — A digital signature that does not conform to the XPS
23 Document signing rules.

24 **digital signature, questionable** — A digital signature that conforms to the XPS Document
25 signing rules but has a problem during validation of the signature such as the inability to
26 contact the certificate authority to validate its authenticity or the markup contains markup
27 compatibility elements and attributes that can change the representation of the signed content.

28 **digital signature, valid** — A digital signature that conforms to the XPS Document signing
29 rules and is not a broken digital signature or questionable digital signature.

30 **document content** — A document structural concept that identifies each block of individually
31 readable content in an XPS Document.

32 **document outline** — A document structural concept that contains a structured index of the
33 content in an XPS Document, much like a table of contents.

34 **driver** — A producer that has specific knowledge of the consumer of the XPS Document.

35 **fixed payload** — A payload that is rooted with a FixedDocumentSequence part.

36 **fixed payload root** — The root of a fixed payload is the FixedDocumentSequence part.

- 1 **FixedDocument part** — A common, easily indexed root for all pages within an XPS Document.
- 2 **FixedDocumentSequence part** — The part that assembles a set of FixedDocument parts
3 within the fixed payload.
- 4 **FixedPage part** — The part that contains all of the visual elements to be rendered on a page.
- 5 **named color** — An industry-defined color specification that identifies a particular color in a
6 well-defined color schema, usually for purposes of printing.
- 7 **named element** — An element in the document structure markup that refers to an element in
8 the fixed-page markup with a specified name.
- 9 **ordering, interleaved**— The layout style of a physical package where parts are broken into
10 pieces and “mixed-in” with pieces from other parts. When delivered, interleaved packages help
11 improve the performance of the consumer processing the package.
- 12 **ordering, simple** — Simple ordering the parts in the package are laid out with a defined
13 ordering. When such a package is delivered in a purely linear fashion, starting with the first
14 byte in the package through to the last that, all of the bytes for the first part arrive first, then
15 all of the bytes for the second part, and so on.
- 16 **package** — A logical entity that holds a collection of parts.
- 17 **package model** — Defines a package abstraction that holds a collection of parts.
- 18 **package relationship** — A relationship whose target is a part and whose source is the
19 package as a whole. Package relationships are found in the package relationships part named
20 “/_rels/.rels”.
- 21 **part** — A stream of bytes with a MIME content type and associated common properties.
22 Typically corresponds to a file (as on a file system), a stream (as in a compound file), or a
23 resource (as in an HTTP URI).
- 24 **part name** — A part name is used to refer to a part in the context of a package, typically as
25 part of a URI. By definition, the part name is the path component of a pack URI.
- 26 **payload** — A complete collection of interdependent parts and relationships within a package.
- 27 **physical imageable size** — Represents the area of a page that is printable by a specific
28 device.
- 29 **physical media size** — Represents the physical media on which the content will be printed.
- 30 **physical model** — Defines the mapping between the components of the package model to the
31 features of a particular physical format based on the ZIP specification.
- 32 **piece** — A portion of a part. Pieces of different parts can be interleaved together. The individual
33 pieces are named using a unique mapping from the part name. Pieces are not addressable in
34 the package model.
- 35 **primary fixed payload root** — The fixed payload root that is referenced by the XPS package
36 StartPart relationship.
- 37 **PrintTicket part** — A PrintTicket part provides the settings used when a package is printed.
38 PrintTicket parts can be attached to the entire package, or at lower levels in the structure, such
39 as individual pages.

- 1 **producer** — A piece of software or a device that writes XPS packages.
- 2 **producer bleed size** — Represents the overflow (or “bleed”) box used by the producer for
3 registration and layout.
- 4 **producer content size** — Represents the content bounding box specified by the producer.
- 5 **producer media size** — Represents the physical media on which the content will be printed.
- 6 **property** — A characteristic of a markup element, referred to as an attribute of the element.
- 7 **property attribute** — An XPS Document property value can be expressed as either a property
8 attribute or a property element.
- 9 **property element** — An XPS Document property value can be expressed as either a property
10 attribute or a property element.
- 11 **property value** — The value of a property, expressed as an XML attribute, an XML child
12 element, or an entry in the resource dictionary.
- 13 **relationships** — A relationship represents the kind of connection between a source part and a
14 target part in a package. Relationships make the connections between parts directly
15 discoverable without looking at the content in the parts, and without altering the parts
16 themselves. See also, package relationship.
- 17 **relationships part** — A part containing an XML representation of relationships.
- 18 **required part** — A part, such as an image or font, that is referenced from other parts, and is
19 required for valid processing of the referencing part.
- 20 **resource definition** — A shareable property value, with a name, defined within a resource
21 dictionary. Any property value defined by fixed page markup can be held in a resource
22 dictionary. Each resource definition has a key that is unique within the scope of the resource
23 dictionary.
- 24 **resource dictionary** — A resource dictionary holds resources. Each resource in a resource
25 dictionary carries a name. The resource’s name can be used to reference the resource from a
26 property’s XML attribute.
- 27 **resource dictionary, remote** — A part containing a resource dictionary.
- 28 **resource reference** — An attribute whose value refers to an entry in a resource dictionary.
29 Resource references appear in the format “{StaticResource RscName}” where RscName
30 corresponds to a matching entry in the resource dictionary with an x:Key attribute value.
- 31 **signature definition** — The means by which XPS Document authors provide co-signature
32 requirements and workflow-specific signature information.
- 33 **signature spot** — A visual element that indicates that a digital signature has been applied or
34 requested.
- 35 **signing rules** — The set of rules that define whether a particular digital signature is compliant
36 with the XPS Document **signature** policy.
- 37 **story** — A block of individually readable content in an XPS Document.
- 38 **story fragment** — A portion of a story that appears within the scope of a single fixed page.

- 1 **stream** — A linearly ordered sequence of bytes.
- 2 **thumbnail** — An images that helps end-users identify parts of a package or a package as a
3 whole.
- 4 **XPS Document** — A package that contains a discoverable fixed payload and is a format for
5 storing paginated documents defined by the XPS specification.
- 6 **XPS Document StartPart relationship** — The specific relationship type that identifies the
7 root of a fixed payload within an XPS Document.
- 8 **ZIP Archive** — A physical ZIP file that is displayed by the file system. A ZIP archive contains
9 **ZIP items**.

5. Notational Conventions

5.1 Document Conventions

Except where otherwise noted, syntax descriptions are expressed in the ABNF format as defined in RFC 4234.



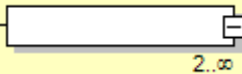
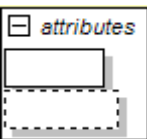


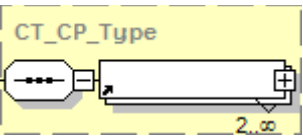
Definition terms are formatted like *this*.

Syntax descriptions and code are formatted in `monospace` type.

Replaceable items are formatted in `monospace cursive` type.

5.2 Diagrams

In some cases, markup semantics are described using diagrams. The diagrams place the parent element on the left, with attributes and child elements to the right. The symbols are described below.

Symbol	Description
	Required element. This box represents an element that MUST appear exactly once in markup when the parent element is included. The "+" and "-" symbols on the right of these boxes have no semantic meaning.
	Optional element. This box represents an element that can appear zero or one times in markup when the parent element is included.
	Range indicator. These numbers indicate that the designated element or choice of elements can appear in markup any number of times within the range specified.
	Attribute group. This box indicates that the enclosed boxes are each attributes of the parent element. Solid-border boxes are required attributes; dashed-border boxes are optional attributes.
	Sequence symbol. The element boxes connected to this symbol can appear in markup in the illustrated sequence only, from top to bottom.
	Choice symbol. Only one of the element boxes connected to this symbol can appear in markup.
	Type indicator. The elements within the dashed box are of the complex type indicated.

1 **6. Acronyms and Abbreviations**

2 The following acronyms and abbreviations are used throughout this specification:

3 IEC — the International Electrotechnical Commission

4 ISO — the International Organization for Standardization

5 W3C — World Wide Web Consortium

1 **7. General Description**

2 This specification is intended for use by implementers, academics, and application
3 programmers. As such, it contains explanatory material that, strictly speaking, is not necessary
4 in a formal specification.

5 This specification is divided into the following subdivisions:

- 6 1. Front matter (clauses 1–7).
- 7 2. XPS Documents (clauses 8–18), which presents the details of the primarily XML-based
8 XPS Document format. These clauses describe the XML markup that defines the
9 composition of documents and the appearance of each page. They also include rendering
10 rules that enable devices and applications to display and print XPS Documents with full
11 fidelity in a wide range of environments and scenarios.
- 12 3. XPS Document Markup Reference (clause 19), which presents a consolidated reference of
13 XPS Document markup elements and their attributes.
- 14 4. Annexes (A–J), which contain additional technical details and schemas, as well as
15 convenient reference information.

16 Examples are provided to illustrate possible forms of the constructions described. References
17 are used to refer to related clauses. Notes are provided to give advice or guidance to
18 implementers or programmers. Annexes provide additional information or summarize the
19 information contained elsewhere in this specification.

20 Clauses 1–5 and 7–19, and annexes A–H and J, form a normative part of this specification; and
21 the clause 6, annex I, examples, notes, and the index, are informative.

22 Except for whole clauses or annexes that are identified as being informative, informative text
23 that is contained within normative text is indicated in the following ways:

- 24 1. Examples within narrative are indicated as follows: [*Example: ... end example*]
- 25 2. Examples of XML are indicated as follows: *Example m.n: caption ... end example*]
- 26 3. [*Note: ... end note*]

8. XPS Document Format

This specification describes how the XPS Document format is organized internally and rendered externally. It is built upon the principles described in the OPC specification. XPS Documents MUST observe all conformance requirements and recommendations of that specification, except where indicated otherwise [M1.1]. The information presented here is intended both for producers, which emit content in the XPS Document format, and consumers, which access and render or process the contents of an XPS Document.

The XPS Document format represents a set of related pages with a fixed layout, which are organized as one or more *documents*, in the traditional meaning of the word. A file that implements this format includes everything necessary to render fully those documents on a display device or physical medium (such as paper). This includes all resources such as fonts and images that might be required to render individual page markings.

In addition, the format includes optional components that build on the minimal set of components required to render a set of pages. This includes the ability to specify print job control instructions, to organize the minimal page markings into larger semantic blocks such as paragraphs, and to rearrange physically the contents of the format for easy consumption in a streaming manner, among others.

Finally, the XPS Document format implements the common package features specified by the OPC specification that support digital signatures and core properties.

8.1 How This Specification Is Organized

This subclause is informative

Clause	Description
Parts and Relationships (§9)	<p>This clause describes how XPS Documents use the packaging model (as described in the OPC specification) to organize data. All part and relationship types are described in detail, including how they are used and what they can contain.</p> <p>This clause also describes the XPS Document markup model, in particular, its parts, and how the XML markup relates to the packaging conventions and recommendations it builds on.</p>
Documents (§10)	<p>The fundamental building blocks of the XPS Document format are described here. This clause describes how pages are composed into larger documents and how documents are composed into document sequences. These components are represented in markup.</p>
Graphics (§11)	<p>This is the first of several clauses that describe page markings, in particular, vector graphics. The concepts of paths, geometries, and figures are introduced. Vector graphics are represented in page-layout XML markup.</p>

Clause	Description
Text (§12)	This clause describes how to include text markings in page-layout markup. It describes how to reference a font and extract information from a font to render the page.
Brushes (§13)	Both vector graphics and text are rendered by applying any of the brushes described in this clause. This includes brushes that are created from solid colors, gradients, images, or other page-layout markup.
Common Properties (§14)	Several page-layout markup elements share a common set of properties. These properties can be expressed either as XML attributes or as XML child or descendant elements. This clause describes these common properties.
Color (§15)	XPS Documents support a wide range of color options and color spaces, both for vector and raster images. This clause describes the combinations of image formats and color markup that can be used. A number of color-related topics are discussed, including color separation, color profiles, and color blending.
Document Structure and Interactivity (§16)	This clause describes the components of the XPS Document format that support assigning larger semantic meaning to individual page markings. [<i>Example: Such markings might be tables or paragraphs. end example</i>] It also provides a mechanism to describe an outline of the document. Additionally, this clause provides guidance on how consumers that enable interactive features such as hyperlinks, selection, and accessibility tools should use the format. It also describes how producers should emit content to enable interactive features.
XPS Document Package Features (§17)	This clause describes how package features (as described in the OPC specification) are used and extended in the XPS Document format. This includes interleaving, digital signatures, and core properties.
Rendering Rules (§18)	This clause provides precise instructions for rendering XPS Document contents to ensure a consistent result among various implementations.
Elements (§19)	The full list of elements described throughout the preceding clauses is assembled in this clause, in alphabetical order, for easy reference.
Signature Definitions Schema (§A)	This annex includes the W3C XSD schema for the Signature Definitions part.
XPS Document Schema (§B)	This annex includes the W3C XSD schema for the FixedDocumentSequence, FixedDocument, and FixedPage parts.
Resource Dictionary Key Schema (§C)	This annex provides the W3C XSD schema for the resource dictionary Key attribute, used by several elements in the XPS Document schema.
Document Structure Schema (§D)	This annex provides the W3C XSD schema for the DocumentStructure and StoryFragments parts.
Discard Control Schema (§E)	This annex includes the W3C XSD schema for the DiscardControl part for interleaving.
Abbreviated Geometry Syntax Algorithm (§F)	This annex provides a sample algorithm for interpreting the abbreviated geometry syntax provided to succinctly describe geometric regions in a single attribute.

Clause	Description
scRGB Gamut Boundary Definition (§G)	The scRGB color space's gamut boundary used by this specification is defined in this annex.
Standard Namespaces and Content Types (§H)	This annex defines all of the XML namespace names, content types, and relationship types used by all XPS Document parts and relationships.
Conformance Requirements (§I)	This annex assembles all the conformance requirements specified throughout the previous clauses and annexes into a comprehensive list for reference purposes.

1 **End of informative text**

2 **8.2 Package**

3 The XPS Document format MUST use a ZIP archive for its *physical model* [M1.2]. The OPC
4 specification describes a packaging model; that is, how the package is represented internally
5 with parts and relationships.

6 The XPS Document format includes a well-defined set of parts and relationships, each fulfilling a
7 particular purpose in the document. The format also extends the package features, including
8 digital signatures, thumbnails, and interleaving.

9 *Figure 8-1. Package-based XPS Document format*



10

1 9. Parts and Relationships

2 The packaging conventions described in the OPC specification can be used to carry any payload.
 3 A *payload* is a complete collection of interdependent parts and relationships within a package.
 4 This specification defines a particular payload that contains a static or “fixed-layout”
 5 representation of paginated content: the fixed payload.

6 A package that holds at least one fixed payload and follows the rules described in this
 7 specification is referred to as an *XPS Document*. Producers and consumers of XPS Documents
 8 can implement their own parsers and rendering engines based on this specification.

9 XPS Documents address the requirements that information workers have for distributing,
 10 archiving, rendering, and processing documents. Using known rendering rules, XPS Documents
 11 can be unambiguously reproduced or printed without tying client devices or applications to
 12 specific operating systems or service libraries. Because the XPS Document is expressed in a
 13 neutral, application-independent way, the content can be viewed and printed without the
 14 application used to create the package.

15 9.1 Fixed Payload

16 A payload that has a FixedDocumentSequence root part is known as a *fixed payload*. A *fixed*
 17 *payload root* is a FixedDocumentSequence part that references FixedDocument parts that, in
 18 turn, reference FixedPage parts.

19 A specific relationship type is defined to identify the root of a fixed payload within an XPS
 20 Document: the *XPS Document StartPart relationship*. The *primary fixed payload root* is the
 21 FixedDocumentSequence part that is referenced by the XPS Document StartPart relationship.
 22 Consumers such as viewers or printers use the XPS Document StartPart relationship to find the
 23 primary fixed payload in a package. The XPS Document StartPart relationship **MUST** point to the
 24 FixedDocumentSequence part that identifies the root of the fixed payload [M2.14].

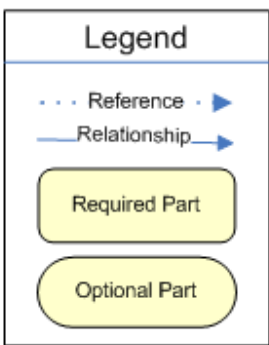
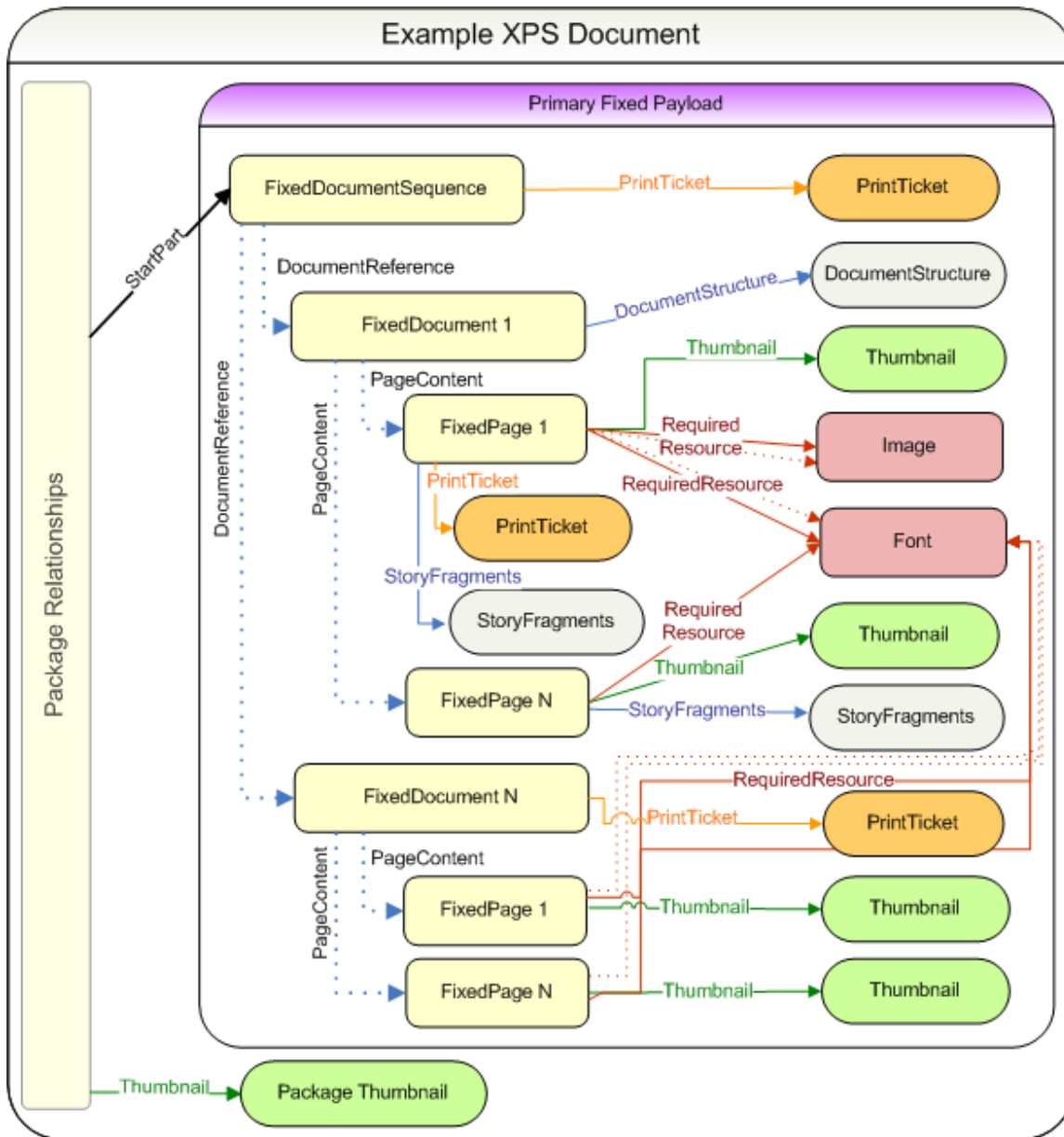
25 The payload includes the full set of parts required for processing the FixedDocumentSequence
 26 part. All content to be rendered **MUST** be contained in the XPS Document [M2.1]. The parts that
 27 can be found in an XPS Document are listed in Table 9–1. Relationships and content types for
 28 these parts are defined in §H. Each part **MUST** use *only* the appropriate content type specified
 29 in §H [M2.2].

30 *Table 9–1. XPS Document parts*

Name	Description	Required/Optional
FixedDocumentSequence (§9.1.2)	Specifies a sequence of fixed documents.	REQUIRED [M2.3]
FixedDocument (§9.1.3)	Specifies a sequence of fixed pages.	REQUIRED [M2.4]
FixedPage (§9.1.4)	Contains the description of the contents of a page.	REQUIRED [M2.5]
Font (§9.1.7)	Contains an OpenType or TrueType font.	REQUIRED if a <Glyphs> element is present [M2.6]

Image (§9.1.5) JPEG image (§9.1.5.1) PNG image (§9.1.5.2) TIFF image (§9.1.5.3) Windows Media Photo image (§9.1.5.4)	References an image file.	REQUIRED if an <ImageBrush> element is present [M2.7]
Remote resource dictionary (§9.1.8)	Contains a resource dictionary for use by fixed page markup.	REQUIRED if a key it defines is referenced [M2.8]
Thumbnail (§9.1.6)	Contains a small JPEG or PNG image that represents the contents of the page or package.	OPTIONAL [O2.1]
PrintTicket (§9.1.9)	Provides settings to be used when printing the package.	OPTIONAL [O2.2]
ICC profile	Contains an ICC Version 2 color profile optionally containing an embedded Windows Color System (WCS) color profile.	OPTIONAL [O2.3]
DocumentStructure (§9.1.11)	Contains the document outline and document contents (story definitions) for the XPS Document.	OPTIONAL [O2.4]
StoryFragments (§9.1.12)	Contains document content structure for a fixed page.	OPTIONAL [O2.5]
SignatureDefinitions (§9.1.10)	Contains a list of digital signature spots and signature requirements.	OPTIONAL [O2.6]
DiscardControl	Contains a list of resources that are safe for consumers to discard during processing.	OPTIONAL [O2.7]

1 Example 9-1. A typical XPS Document



2
3 end example]

9.1.1 Fixed Payload Relationships

Internal resources are associated with parts by relationships and inline references. XPS Documents MUST NOT reference external XPS resources [M2.1]. In general, inline resource references are represented inside the referring part in ways that are specific to the content type of the part, that is, in arbitrary markup or application-specific encoding. Relationships represent the type of connection between a source part and a target resource, and they allow parts to be related without modifying them. For more information, see the OPC specification.

Resources, which include fonts, images, color profiles, and remote resource dictionaries, that are referenced by inline URIs but are necessary to render the page MUST use the Required Resource relationship from the FixedPage part to the resource [M2.10]. If any resource references *other* resources, the producer MUST also use the Required Resource relationship from the FixedPage part to the indirectly referenced resource [M2.10].

It is RECOMMENDED that there be exactly *one* Required Resource relationship from the FixedPage part for each resource referenced from markup [S2.1]. Multiple Required Resource relationships from a FixedPage part to a resource are not considered an error, but they reduce efficiency. It is not considered an error if a FixedPage part that does not use a specific resource in its markup references the resource via a Required Resource relationship; however, doing so might reduce efficiency for consumers.

Relationship types are defined in §H.

Table 9–2. Fixed payload relationships

Name	Description	Required/Optional
Core Properties	Relationship from the package to the Core Properties part.	OPTIONAL [O2.8]
Digital Signature Origin	Relationship from the package to the Digital Signature Origin part.	OPTIONAL [O2.9]
Digital Signature	Relationship from the Digital Signature Origin part to an Digital Signature XML Signature part.	OPTIONAL [O2.10]
Digital Signature Certificate	Relationship from a Digital Signature XML Signature part to a Digital Signature Certificate part.	OPTIONAL [O2.11]
Digital Signature Definitions	Relationship from the FixedDocument part to a Digital Signature Definitions part.	OPTIONAL [O2.12]
DiscardControl	Relationship from the package to a DiscardControl part.	OPTIONAL [O2.13]
DocumentStructure	Relationship from the FixedDocument part to a DocumentStructure part.	OPTIONAL [O2.14]
PrintTicket	Relationship from a FixedDocumentSequence part, a FixedDocument part, or a FixedPage part to a PrintTicket part.	OPTIONAL [O2.15]
Required Resource	Relationship from a FixedPage part to a required resource, including	REQUIRED for each resource referenced

	Font, Image, ColorProfile, and Remote Resource Dictionary parts. Required resources can be shared between pages.	from a FixedPage [M2.10]
Restricted Font	Relationship from a FixedDocument part to a Font part. Specifies the referenced font as restricted, disallowing any modification or editing of any <Glyphs> element text using the referenced font.	REQUIRED for each preview and print font used [M2.12]
StartPart	Relationship from the package to the FixedDocumentSequence part that is the primary fixed payload root.	REQUIRED [M2.13, M2.14]
StoryFragments	Relationship from a FixedPage part to the StoryFragments part for the page.	OPTIONAL [O2.16]
Thumbnail	Relationship from the package to an Image part or from a FixedPage part to an Image part.	OPTIONAL [O2.17]

1 Producers that generate a relationship MUST include the target part in the XPS Document for
 2 any of the following relationship types: DiscardControl, DocumentStructure, PrintTicket,
 3 Required Resource, Restricted Font, StartPart, StoryFragments, and Thumbnail. Consumers
 4 that access the target part of any relationship with one of these relationship types MUST
 5 generate an error if the part is not included in the XPS Document [M2.77].

6 **9.1.2 FixedDocumentSequence Part**

7 The *FixedDocumentSequence part* assembles a set of fixed documents within the fixed payload.
 8 [Example: A printing client can assemble two separate documents, a two-page cover memo and
 9 a twenty-page report (both are FixedDocument parts), into a single package to send to the
 10 printer. end example]

11 The FixedDocumentSequence part is the only valid root of a fixed payload. Even if an XPS
 12 Document contains only a single fixed document, the FixedDocumentSequence part is still used.
 13 One FixedDocumentSequence part per fixed payload is REQUIRED [M2.3].

14 Fixed document sequence markup specifies each fixed document in the fixed payload in
 15 sequence, using <DocumentReference> elements. The order of <DocumentReference>
 16 elements determines document order and MUST be preserved by consumers that are also
 17 producers [M2.15]. Each <DocumentReference> element MUST reference a FixedDocument
 18 part by relative URI [M2.80]. For more information, see §10.1.

19 The content type of the FixedDocumentSequence part is defined in §H.

20 **9.1.3 FixedDocument Part**

21 The *FixedDocument part* is a common, easily indexed root for all pages within the document. A
 22 fixed document identifies the set of fixed pages for the document.

23 The markup in the FixedDocument part specifies the pages of a document in sequence using
 24 <PageContent> elements. The order of <PageContent> elements determines page order and

1 MUST be preserved by consumers that are also producers [M2.16]. Each <PageContent>
 2 element MUST reference a FixedPage part by relative URI [M2.81]. For more information,
 3 see §10.2.

4 The content type of the FixedDocument part is defined in §H,

5 **9.1.4 FixedPage Part**

6 The *FixedPage part* contains all of the visual elements to be rendered on a page. Each page has
 7 a fixed size and orientation. The layout of the visual elements on a page is determined by the
 8 fixed page markup. This applies to both graphics and text, which are represented with precise
 9 typographic placement. The contents of a page are described using a powerful but simple set of
 10 visual primitives.

11 Each FixedPage part specifies the contents of a page within a <FixedPage> element using
 12 <Path> and <Glyphs> elements (using various brush elements) and the <Canvas> grouping
 13 element. The <ImageBrush> and <Glyphs> elements (or their child or descendant elements)
 14 can reference Image parts or Font parts by URI. They MUST reference these parts by relative
 15 URI [M2.82]. For more information, see §10.3.

16 The content type of the FixedPage part is defined in §H

17 **9.1.5 Image Parts**

18 Image parts reference image files. A single image can be shared among multiple fixed pages in
 19 one or more fixed documents. Images referenced in markup MUST be internal to the package
 20 [M2.1]. References to images that are external to the package are invalid.

21 Images are included in XPS Documents with an <ImageBrush> element and an ImageSource
 22 attribute to reference a part with the appropriate content type. For more information, see §H.2.
 23 Fixed pages MUST use a Required Resource relationship to each Image part referenced
 24 [M2.10]. For more information, see §H.3.

25 XPS Documents support the following image formats:

- 26 • JPEG
- 27 • PNG
- 28 • TIFF
- 29 • Windows Media Photo

30 Color profiles MAY be embedded in image files [O2.18] ~~and SHOULD be used by consumers~~
 31 ~~[S2.5]~~. See §15.

32 **9.1.5.1 JPEG Images**

33 It is RECOMMENDED that JPEG image part names end with the extension “.jpg” [S2.6]. JPEG
 34 image parts MUST contain images that conform to the JPEG specification [M2.17]. Consumers
 35 SHOULD support JPEG images that contain JFIF-specified APP0 and ICC-specified APP2 markers
 36 [S2.34]. Consumers MUST support JPEG images that contain the EXIF-specified APP1 marker
 37 and interpret the EXIF color space correctly [M2.78].

38 *Table 9–3. Supported JPEG APPn markers*

APPn marker	Originating source
APP0	JFIF specification

APP1	EXIF extension defined by JEITA
APP2	ICC profile marker defined by the ICC specification

1 [Note: Implementers of consumers might wish to support additional APPn markers, such as
 2 APP13 (Photoshop 3.0 extension) and APP14 (Adobe DCT Filters in PostScript Level 2
 3 extension). *end note*]

4 [In cases where a consumer encounters a JPEG image with conflicting resolution information in
 5 different markers, the order of precedence is as follows:](#)

- 6 [1. The EXIF tag](#)
- 7 [2. The JFIF tag](#)
- 8 [3. Any other APPn tags supported by the consumer](#)
- 9 [4. A default value of 96 dpi \(as described in §13.4.1\)](#)

10 Some JPEG implementations have limited support for CMYK JPEG images, such as:

- 11 • CMYK is converted to RGB in the decoder using fixed tables instead of the supplied ICC
 12 profile.
- 13 • ICC Profiles embedded using APP2 are limited in length, because APPn marker chunking
 14 is not supported.

15 Therefore, the use of JPEG CMYK images is NOT RECOMMENDED in XPS Documents because
 16 rendering results can differ significantly between implementations. TIFF or Windows Media
 17 Photo images SHOULD be used instead to represent CMYK images [S2.7].

18 ~~[Note: If both [ICC-specified](#) APP2, and APP13 markers are specified, the [ICC-specified](#) APP2
 19 marker takes precedence. *end note*]~~ If the JPEG image is embedded in a TIFF image, the TIFF
 20 ICC profile settings are used.

21 If no color profile is embedded in the JPEG image or stored in a separate part associated with
 22 the JPEG image according to the mechanisms described in §15.3.8, then the default color space
 23 MUST be treated as defined in §15.3.9 [M8.30].

24 **9.1.5.2 PNG Images**

25 It is RECOMMENDED that PNG image part names end with the extension “.png” [S2.8]. PNG
 26 image parts MUST contain images that conform to the PNG specification [M2.18].

27 *Table 9–4. Support for ancillary PNG chunks*

Chunk	Support Level
tRNS	MUST Support [M2.19]
iCCP	MUST Support [M2.20]
sRGB	MUST Ignore [M2.21]
cHRM	MUST Ignore [M2.22]
gAMA	MUST Ignore [M2.23]
sBIT	MUST Ignore [M2.24]

1 If no color profile is embedded in the PNG image or stored in a separate part associated with
 2 the PNG image according to the mechanisms described in §15.3.8, then the default color space
 3 MUST be treated as defined in §15.3.9 [M8.30].

4 **9.1.5.3 TIFF Images**

5 It is RECOMMENDED that TIFF image part names end with the extension “.tif” [S2.9]. TIFF
 6 image parts MUST contain images that conform to the TIFF specification [M2.25]. XPS
 7 Document consumers MUST support baseline TIFF 6.0 with some extensions, as noted in Table
 8 9–5 [M2.26]. These tags MUST be supported for the specified image types [M2.26]. If
 9 consumers encounter a tag that is not included below, they SHOULD ignore that tag [S2.10].

10 *Table 9–5. Supported TIFF-6.0 tags*

Image type	Tags
Bilevel images	PhotometricInterpretation (0 and 1) Compression (1, 2, 3, 4, 5, or 32773) ImageLength ImageWidth ResolutionUnit (1, 2, or 3) RowsPerStrip StripByteCounts StripOffsets XResolution YResolution
Grayscale images	PhotometricInterpretation (0 and 1) BitsPerSample (4, 8, or 16) Compression (1, 5, 7 6, or 32773) ImageLength ImageWidth ResolutionUnit (1, 2, or 3) RowsPerStrip StripByteCounts StripOffsets XResolution YResolution
Palette color images	BitsPerSample (1, 4, or 8) ColorMap Compression (1, 5, or 32773) ImageLength ImageWidth PhotometricInterpretation (3) ResolutionUnit (1, 2, or 3) RowsPerStrip StripByteCounts StripOffsets

	XResolution
	YResolution
RGB images	<p>BitsPerSample (8,8,8 or 16,16,16; or if SamplesPerPixel = 4: ExtraSamples=1 or 2: 8,8,8,8 or 16,16,16,16)</p> <p>Compression (1, 5, 76, or 32773)</p> <p>ExtraSamples (0, 1, or 2. Required if SamplesPerPixel = 4; must not be present otherwise)</p> <p>ICC Color Profile [tag 34675]</p> <p>ImageLength</p> <p>ImageWidth</p> <p>PhotometricInterpretation (2)</p> <p>PlanarConfiguration (1)</p> <p>ResolutionUnit (1, 2, or 3)</p> <p>RowsPerStrip</p> <p>SamplesPerPixel (3 or 4; or if ExtraSamples=1 or 2: 4)</p> <p>StripByteCounts</p> <p>StripOffsets</p> <p>XResolution</p> <p>YResolution</p>
CMYK images (TIFF extension)	<p>BitsPerSample (8,8,8,8 or 16,16,16,16; or if SamplesPerPixel = 5: ExtraSamples=1 or 2: 8,8,8,8,8 or 16,16,16,16,16)</p> <p>Compression (1, 5, 76, or 32773)</p> <p>ExtraSamples (0, 1, or 2. Required if SamplesPerPixel = 5; must not be present otherwise)</p> <p>ICC Color Profile [tag 34675]</p> <p>ImageLength</p> <p>ImageWidth</p> <p>InkSet (1)</p> <p>NumberOfInks (4)</p> <p>PhotometricInterpretation (5)</p> <p>PlanarConfiguration (1)</p> <p>ResolutionUnit (1, 2, or 3)</p> <p>RowsPerStrip</p> <p>SamplesPerPixel (4 or 5; or if ExtraSamples=1 or 2: 5)</p> <p>StripByteCounts</p> <p>StripOffsets</p> <p>XResolution</p> <p>YResolution</p>

- 1 If the TIFF image contains multiple image file directories (IFDs), consumers MUST use only the first IFD and ignore all others [M2.27].
- 2

- 1 If the ResolutionUnit tag is set to 1 (no units), XResolution and YResolution are interpreted in
2 the same manner as if the ResolutionUnit was set to 2 (inches).
- 3 If no color profile is embedded in the TIFF image or stored in a separate part associated with
4 the TIFF image according to the mechanisms described in §15.3.8, then the default color space
5 MUST be treated as defined in §15.3.9 [M8.30].
- 6 The following features of the TIFF specification MUST be supported in addition to the tags
7 described in Table 9–5:
- 8 • Baseline TIFF (Sections 1–10) with the exception of the following tags [M2.26]:
 - 9 ○ CellLength
 - 10 ○ CellWidth
 - 11 ○ GrayResponseCurve
 - 12 ○ GrayResponseUnit
 - 13 ○ MaxSampleValue
 - 14 ○ MinSampleValue
 - 15 ○ Orientation
 - 16 ○ Thresholding
 - 17 • CCITT bilevel encodings (Section 11) [M2.28]
 - 18 • CMYK images (Section 16) [M2.29]

- 1 • Associated alpha data (Section 18) [M2.30]
 - 2 ○ [ExtraSamples tag value of 0: The data in this channel MUST be ignored \[M2.83\]](#)
 - 3 ○ ExtraSamples tag value of 1: Treat alpha as pre-multiplied alpha (see §18.4.1 for
 - 4 details)
 - 5 ○ ExtraSamples tag value of 2: Treat alpha as non-pre-multiplied alpha
- 6 • LZW compression (Section 13) [M2.31]
- 7 • Differencing predictors (Section 14) [M2.32]
- 8 • JPEG compression (Section 22)
 - 9 ○ Only compression mode 6 MUST be supported [M2.33]
- 10 • Embedded ICC Profile (described in the ICC specification) [M2.34]
- 11 • EXIF IFD (tag 34665) as described in the EXIF specification. The EXIF color space MUST
- 12 be interpreted correctly [M2.79].

13 Consumers that support tags and features not described above can result in undesirable
 14 differences in the appearance of XPS Documents. Producers cannot rely on a consistent
 15 interpretation of tags or features that are not described above and therefore SHOULD NOT use
 16 any such tags or features [S2.10].

17 ~~[Note: Many TIFF images in circulation today deviate from the TIFF Specification. Over time,~~
 18 ~~TIFF-consuming implementations have developed a certain tolerance for such deviations by~~
 19 ~~attempting to deduce the intent of the TIFF image author and correct for apparent errors or~~
 20 ~~deviations.~~

21 ~~Therefore,~~ XPS Document consumers should [mitigate the effect of badly formed TIFF files in the](#)
 22 ~~following ways do the following~~ [S2.11]:

- 23 • Test with as many different TIFF images as possible.
- 24 • Correct common mistakes in TIFF images, such as:
 - 25 ○ Not all BitsPerSample hold the same value
 - 26 ○ Number of BitsPerSample does not match SamplesPerPixel
 - 27 ○ PhotometricInterpretation 1 or 2 (instead of 3) used when BitsPerSample is set to
 - 28 "8,8,8"
 - 29 ○ ~~Missing ExtraSamples tag, but SamplesPerPixel does not match the value expected~~
 30 ~~by the PhotometricInterpretation tag, thus necessitating deduction of the~~
 31 ~~ExtraSamples value. When the ExtraSamples tag is missing and SamplesPerPixel is~~
 32 ~~not consistent with the PhotometricInterpretation tag then ExtraSamples values~~
 33 ~~should be given the value 0.~~
- 34 • Implement a recovery strategy when a problematic TIFF image is encountered.

35 [Many TIFF images in circulation today deviate from the TIFF Specification.](#) *end note*]

36 9.1.5.4 Windows Media Photo Images

37 It is RECOMMENDED that Windows Media Photo image part names end with the extension
 38 ".wdp" [S2.12]. Windows Media Photo image files MUST conform to the Windows Media Photo
 39 specification [M2.35]. XPS Documents support Windows Media Photo images with the
 40 characteristics identified in Table 9–6.

1 *Table 9–6. Supported Windows Media Photo features*

Color space	Pixel formats	Compression	Alpha
Grayscale	8-bit integer	Lossy	None
	16-bit integer	Lossless	
	16-bit half-float*		
	16-bit fixed point*		
	32-bit fixed point*		
sRGB	8-bit integer	Lossy	1-channel
	16-bit integer	Lossless	1-channel pre-multiplied
scRGB	16-bit half-float	Lossy	1-channel
	16-bit fixed point	Lossless	1-channel pre-multiplied
	32-bit IEEE float		RGBE-Radiance (no alpha channel)
	32-bit fixed point RGBE-Radiance		
CMYK	8-bit integer	Lossy	1-channel independent
	16-bit integer	Lossless	
N-channel (including named color N-tone)	8-bit integer	Lossy	1-channel independent
	16-bit integer	Lossless	
Profiled RGB (3-channel)	8-bit integer	Lossy	1-channel
	16-bit integer	Lossless	1-channel pre-multiplied

2 * The value range of these formats is the same as scRGB.

3 If no color profile is embedded in the Windows Media Photo image or stored in a separate part
 4 associated with the Windows Media Photo image according to the mechanisms described
 5 in §15.3.8, then the default color space MUST be treated as defined in §15.3.9 [M8.30].

6 **9.1.6 Thumbnail Parts**

7 Thumbnails are small images that represent the contents of a fixed page or an entire XPS
 8 Document. Thumbnails enable users of viewing applications to select a page easily.

9 Thumbnail images MAY be attached using a relationship to the FixedPage parts [O2.19]. Each
 10 FixedPage part MUST NOT have more than one thumbnail part attached [M2.36]. Relationships
 11 to thumbnail parts are defined in §H. It is RECOMMENDED that if thumbnails are used for
 12 pages, a thumbnail SHOULD be included for each page in the document [S2.13].

13 Although the OPC specification allows thumbnails to be attached to any part, XPS Document
 14 consumers SHOULD only process thumbnails associated via a package relationship from the
 15 package as a whole or via a relationship from a FixedPage part [S2.14]. These thumbnails
 16 MUST be in either JPEG or PNG format [M2.37]. Thumbnails attached to any other part SHOULD
 17 be ignored by XPS Document consumers [S2.14]. The content types of thumbnail parts are
 18 specified in §H.2.

19 For more information about the relationship type for thumbnail parts, see §H.3.

9.1.7 Font Parts

Fonts are stored in font parts. XPS Documents MUST support the OpenType font format, which includes TrueType and CFF fonts [M2.39]. To support portability, Unicode-encoded fonts SHOULD be used (see §9.1.7.5 for additional information) [S2.15].

Font parts are referenced using the `FontUri` attribute of the `<Glyphs>` element. A single font can be shared among multiple fixed pages in one or more fixed documents. Font references MUST be internal to the package; external references to fonts are invalid [M2.1].

If the referenced font part is a TrueType Collection, the fragment portion of the URI indicates the font face to be used. The use of URI fragments is specified in the BNF of Generic URI Syntax specification. The fragment contained in the `FontURI` attribute value MUST be an integer between 0 and $n-1$, inclusive, where n is the number of font faces contained in the TrueType Collection [M2.38]. The syntax for the integer value is expressed as:

```
fontface = *DIGIT
```

[Example: To reference the first font face in the font part `../Resources/Fonts/CJKSuper.ttc`, the value of the `FontUri` attribute is `../Resources/Fonts/CJKSuper.ttc#0`. end example] If no fragment is specified, the first font face is used in the same way as if the URI had specified `#0`. If the fragment is not recognised as a valid integer, consumers SHOULD generate an error [S2.35].

Content types for fonts differ depending on whether the font is non-obfuscated or obfuscated (see §9.1.7.2). Content types are summarized in §H.

Fixed pages MUST use a Required Resource relationship to each Font parts referenced [M2.10]. For more information, see §H.

9.1.7.1 Subsetting Fonts

XPS Documents represent text using the `<Glyphs>` element. Since the format is fixed, it is possible to create a font subset that contains only the glyphs required by the package. Fonts MAY be subsetted based on glyph usage [O2.20]. Although a subsetted font does not contain all the glyphs in the original font, it MUST be a valid OpenType font file [M2.39]. Requirements for valid OpenType font files are described in the OpenType Font File specification.

9.1.7.2 OpenType Font Embedding

Protecting the intellectual property of font vendors is a goal of the XPS Document format. Therefore, producers MUST observe the guidelines and mechanisms described below in order to honor the licensing rights specified in OpenType fonts [M2.40]. It is not the responsibility of consumers to enforce font licensing intent, although consumers MUST be able to process XPS Documents using any combination of these embedding and obfuscation mechanisms, even if produced in violation of these guidelines [M2.41].

The licensing rights of an OpenType font are specified in the `fsType` field of the required OS/2 table in the font file. Table 9–7 lists the bit mask values that can appear in arbitrary combinations in the `fsType` field. Also listed are short descriptions of the licensing right intents and requirements or recommendations. These requirements represent the “rules” that producers and consumers must follow in order to respect licensing rights specified in the font.

For further details on licensing rights of OpenType fonts, see the description of the OS/2 table in “OS/2 and Windows Metrics.”

1 Table 9–7. Guidelines for OpenType font embedding

Bit/mask	Licensing right intent	Producer rules	Consumer rules
– / 0x0000	Installable embedding.	SHOULD do embedded font obfuscation [S2.16] (see §9.1.7.3 for details).	SHOULD NOT extract or install permanently (see below) [S2.17].
0 / 0x0001	Reserved, must be 0.		
1 / 0x0002	Restricted license embedding. If <i>only</i> this bit is set, the font MUST NOT be modified, embedded or exchanged in any manner without obtaining permission from the legal owner.	MUST NOT embed [M2.42]. SHOULD generate a path filled with an image brush referencing an image of rendered characters [S2.18]. SHOULD include the text in the AutomationProperties.Name attribute of the <Path> element [S2.18].	Render embedded images.
2 / 0x0004	For preview and print embedding, font can be embedded and temporarily used on remote systems. However, documents containing <i>any</i> preview and print fonts MUST NOT be modified or edited [M2.43].	MUST do embedded font obfuscation [M2.44] (see §9.1.7.3). MUST add a Restricted Font relationship to the FixedDocument part of the document containing the font [M2.12]. See §12.1.7 and §H.3 for details.	MUST NOT extract or install permanently [M2.45]. MUST NOT modify or edit the XPS Document markup or hierarchical structure starting from the <FixedDocument> element [M2.43].
3 / 0x0008	Editable embedding.	MUST do embedded font obfuscation [M2.46] (see §9.1.7.3).	MUST NOT extract or install permanently [M2.47].
4–7	Reserved, must be 0.		
8 / 0x0100	No subsetting.	MUST do embedded font obfuscation (see §9.1.7.3) [M2.48]. MUST NOT subset font before embedding. [M2.49]	MUST NOT extract or install permanently [M2.50].
9 / 0x0200	Bitmap embedding only.	MUST do embedded font obfuscation [M2.51] (see §9.1.7.3). MUST embed <i>only</i> bitmap characters contained in the font [M2.51]. If no bitmap characters	MUST NOT extract or install permanently [M2.52].

are present in the font,
MUST NOT embed the
font [M2.51].

10–15 Reserved, must be 0.

1 **9.1.7.3 Embedded Font Obfuscation**

2 Embedded font obfuscation is a means of preventing casual misappropriation of embedded
3 fonts. Specifically, embedded font obfuscation prevents end-users from using standard ZIP
4 utilities to extract fonts from XPS Document files and install them on their systems.

5 Embedded font obfuscation is *not* considered a strong encryption of the font data.

6 Embedded font obfuscation achieves the following goals:

- 7 1. Obfuscated font files are embedded within an XPS Document package in a form that
8 cannot be directly installed on any client operating system.
- 9 2. Obfuscated font files are closely tied to the content referencing them. Therefore, it is
10 non-trivial to misappropriate fonts by moving them from one package to another.
- 11 3. The manner in which obfuscated font files are tied to the content referencing them still
12 allows for document merging.

13 For information on how to determine when fonts must be obfuscated prior to embedding, see
14 Table 9–7. above.

15 Although the licensing intent allows embedding of non-obfuscated fonts and installation of the
16 font on a remote client system under certain conditions, this is NOT RECOMMENDED in XPS
17 Documents [S2.19]. However, there are vertical solutions in which implementations might
18 benefit from un-obfuscated font embedding. In these cases, implementations could omit
19 obfuscation or extract and install the embedded font.

20 If a producer is required to perform embedded font obfuscation, it MUST satisfy the following
21 requirements [M2.53]:

- 22 1. Generate a 128-bit GUID (Globally Unique Identifier) for the font to be obfuscated.
23 Instead of a true GUID, a 128-bit random number MAY be used [O2.21]. The 16 bytes of
24 the 128-bit GUID are referred to in the following text by the placeholder names B_{00} , B_{01} ,
25 B_{02} , B_{03} ; B_{10} , B_{11} ; B_{20} , B_{21} ; B_{30} , B_{31} , B_{32} , B_{33} , B_{34} , B_{35} , B_{36} , and B_{37} . The order in which
26 bytes are assigned to these placeholders does not matter, as long as it is consistent for
27 obfuscation and de-obfuscation.
- 28 2. Generate a part name for the obfuscated font using the GUID. The last segment of the
29 part name MUST be of the form " $B_{03}B_{02}B_{01}B_{00}-B_{11}B_{10}-B_{21}B_{20}-B_{30}B_{31}-B_{32}B_{33}B_{34}B_{35}B_{36}B_{37}$ " or
30 " $B_{03}B_{02}B_{01}B_{00}-B_{11}B_{10}-B_{21}B_{20}-B_{30}B_{31}-B_{32}B_{33}B_{34}B_{35}B_{36}B_{37}.ext$ " where each B_x represents a
31 placeholder for one byte of the GUID, represented as two hex digits [M2.54]. The part
32 name MAY have an arbitrary extension (identified by the placeholder ".ext") [O2.22]. It is
33 RECOMMENDED that the extension for TrueType fonts be ".odttf" and for TrueType
34 collections be ".odttc" [S2.20].
- 35 3. The content type for the part containing the obfuscated font MUST match the definition
36 in §H [M2.2].
- 37 4. Perform an XOR operation on the first 32 bytes of the binary data of the font part with
38 the array consisting of the bytes referred to by the placeholders B_{37} , B_{36} , B_{35} , B_{34} , B_{33} ,

1 B_{32} , B_{31} , B_{30} , B_{20} , B_{21} , B_{10} , B_{11} , B_{00} , B_{01} , B_{02} , and B_{03} , in that order and repeating the array
2 once. The result is an obfuscated font.

3 5. Store the obfuscated font in a part with the generated name.

4 When processing fonts, consumers MUST follow these steps [M2.53]:

- 5 1. If the content type of the part containing the font is not the obfuscated font content type
6 as specified in H, process the font without any de-obfuscation steps.
- 7 2. For font parts with the obfuscated font content type as specified in H, de-obfuscate the
8 font by following these rules:[0-2](#).
 - 9 a. Remove the extension from the last segment of the name of the part containing the
10 font.
 - 11 b. Convert the remaining characters of the last segment to a GUID using the byte
12 ordering described above.
 - 13 c. Perform an XOR operation on the first 32 bytes of the binary data of the obfuscated
14 font part with the array consisting of the bytes referred to by the placeholders B_{37} ,
15 B_{36} , B_{35} , B_{34} , B_{33} , B_{32} , B_{31} , B_{30} , B_{20} , B_{21} , B_{10} , B_{11} , B_{00} , B_{01} , B_{02} , and B_{03} , in that order
16 and repeating the array once. The result is a non-obfuscated font.
 - 17 d. Use the non-obfuscated font for the duration of the document processing, but do not
18 leave any local or otherwise user-accessible copy of the non-obfuscated font.

19 **9.1.7.4 Print and Preview Restricted Fonts**

20 If a producer embeds a font with the print and preview restriction bit set, it MUST also add a
21 Restricted Font relationship from the FixedDocument part that includes the FixedPage
22 referencing the font to the restricted font [M2.12].

23 Consumers that are also producers MUST NOT edit a document where the FixedDocument part
24 has a Restricted Font relationship [M2.43]. When invoking editing functionality, consumers that
25 are also producers MUST treat as an error any font with the print and preview restriction bit set
26 for which no Restricted Font relationship has been added to the FixedDocument part [M2.12].

27 Consumers that are not also producers MUST consider an XPS Document valid even if the
28 producer failed to properly set the Restricted Font relationship [M2.12].

29 **9.1.7.5 Non-Standard Font Compatibility Encoding**

30 When processing <Glyphs> elements, the consumer MUST first select a cmap table from the
31 OpenType font following the order of preference shown below (highest listed first) [M2.55]:

32 *Table 9-8. Cmap table selection*

Platform ID	Encoding ID	Description
3	10	Unicode with surrogates
3	1	Unicode without surrogates
3	5	Wansung
3	4	Big5
3	3	Prc
3	2	ShiftJis
3	0	Symbol

0	Any	Unicode (deprecated)
1	0	MacRoman

- 1 All further processing for that font MUST use the selected cmap table [M2.55].
- 2 If a Wansung, Big5, Prc, ShiftJis or MacRoman cmap has been selected, the consumer MUST
3 correctly map from Unicode codepoints in the UnicodeString to the corresponding codepoints
4 used by the cmap before looking up the glyphs [M2.56]. The Unicode standard provides details
5 of the required mappings.
- 6 Producers SHOULD avoid using fonts lacking a Unicode-encoded cmap table [S2.15].
- 7 When processing <Glyphs> elements that reference a cmap (3,0) encoding font, consumers
8 MUST be prepared for the case in which the UnicodeString attribute contains character codes
9 instead of PUA codepoints [M2.57]. This condition is indicated by an unsuccessful Unicode
10 lookup of the codepoint specified in the Unicode string in the cmap (3,0) table. In this case, the
11 correct glyph index is computed by following the general recommendations of the OpenType
12 specification.
- 13 When processing <Glyphs> elements that use this compatibility encoding, character codes in
14 the range 0x20-0xff are mapped to PUA codepoints. Therefore, character codes in the range
15 0x80-0x9f are not considered non-printable Unicode control codes.
- 16 This non-standard encoding has been included to facilitate document production for certain
17 producers. However, there are significant drawbacks resulting from this encoding:
- 18 • Search is unpredictable
 - 19 • Copy and paste functionality is unpredictable
- 20 Producers SHOULD NOT use this non-standard encoding and they SHOULD write PUA
21 codepoints to the UnicodeString attribute [S2.15].

22 **9.1.8 Remote Resource Dictionary Parts**

23 A *remote resource dictionary* allows producers to define resources that can be reused across
24 many pages, such as a brush. This is stored in a Remote Resource Dictionary part. For more
25 information, see §14.2.3.1.

26 **9.1.9 PrintTicket Parts**

27 *PrintTicket parts* provide user intent and device configuration information to printing
28 consumers. PrintTicket parts MUST be processed when the XPS Document is printed [M2.58].
29 PrintTicket parts can be attached only to FixedDocumentSequence, FixedDocument and
30 FixedPage parts and each of these parts MUST attach no more than one PrintTicket [M2.59].
31 PrintTickets can provide override settings to be used when printing the part to which they are
32 attached.

33 **9.1.9.1 PrintTicket Format**

34 The PrintTicket is XML that provides print settings in a consistent, accessible, and extensible
35 manner. Valid PrintTicket settings are specified in the Print Schema. Within the context of an
36 XPS Document, the PrintTicket is generated by the producer. Producers should note that an XPS
37 Document might be printed on various devices, and that the settings included in the PrintTicket
38 SHOULD support portability [S2.21]. Producers and consumers should note that not all
39 PrintTicket keywords defined in the Print Schema are applicable to XPS Documents.

1 9.1.9.2 Mapping PrintTicket Parts to Fixed Payload Parts

2 The PrintTicket defines a hierarchy of print settings to identify the applicability of a setting to
3 different content levels within a print job. Specifically, the PrintTicket supports a hierarchy that
4 is rooted in the print job. Multiple documents are derived from the print job, and multiple pages
5 are defined from each document. A PrintTicket can be associated with each level in this
6 hierarchy. These PrintTicket parts are labeled "job-level," "document-level," and "page-level,"
7 respectively.

8 Print settings are defined within the PrintTicket. Each print setting has a scoping prefix to
9 indicate the level at which it applies. [*Example*: "JobInputBin" is a job-level setting,
10 "DocumentStaple" is a document-level setting, and "PageOrientation" is a page-level setting.
11 *end example*] Settings in level-specific PrintTicket parts are restricted by the scoping prefix. A
12 level-specific PrintTicket MUST contain only settings scoped to the current level and child levels
13 [M2.59]. Job-level PrintTicket parts MUST contain only job-, document-, and page-scoped
14 settings; document-level PrintTicket parts MUST contain only document-scoped and page-
15 scoped settings; and page-level PrintTicket parts MUST contain only page-scoped settings
16 [M2.59].

17 Within an XPS Document, there is a direct mapping between the PrintTicket levels and the fixed
18 payload parts:

- 19 • Consumers MUST process job-level, document-level and page-level settings of PrintTicket
20 parts associated with FixedDocumentSequence parts [M2.60].
- 21 • Producers SHOULD only attach PrintTicket parts containing only document-level and
22 page-level settings with FixedDocument parts [S2.22].
- 23 • Consumers MUST process document-level and page-level settings of PrintTicket parts
24 associated with FixedDocument parts and MUST ignore job-level settings of PrintTicket
25 parts associated with FixedDocument parts [M2.61].
- 26 • Producers SHOULD only attach PrintTicket parts containing only page-level settings with
27 FixedPage parts [S2.23].
- 28 • Consumers MUST process page-level settings of PrintTicket parts associated with
29 FixedPage parts and MUST ignore job-level and document-level settings of PrintTicket
30 parts associated with FixedPage parts [M2.62].

31 PrintTicket parts are associated with parts via the PrintTicket relationship defined in §H.

32 9.1.9.3 Processing PrintTicket Parts

33 Printing consumers MUST process all PrintTicket parts within the XPS Document [M2.58].

34 When processing a PrintTicket, consumers MUST first remove all levels of PrintTicket content
35 not applicable to the current element [M2.63] (see §9.1.9.2).

36 Second, consumers MUST validate the PrintTicket according to the methods defined in the
37 PrintTicket Validation Checklist of the Print Schema documentation [M2.64]. Following
38 validation, the printing consumer MUST properly interpret the print settings according to the
39 following rules for merging two PrintTicket parts [M2.65].

1 Print settings are expressed by scoped Print Schema elements. Elements can interact between
 2 different levels in the PrintTicket hierarchy. Elements that interact between levels MUST be
 3 specified at the root of each level ticket [M2.59]. A keyword merge conflict between PrintTicket
 4 settings is defined as the same root-level Print Schema element denoted by the same name
 5 attribute value appearing in multiple level tickets. There are two options for interactions:

6 ~~1.3.~~ 3. If there is no merge conflict, a prefix-scoped element MUST be pushed down, or
 7 inherited, from a more general ticket to a more specific ticket [M2.66]. This case is
 8 isomorphic to the case where both tickets contain an identical element.

9 ~~2.4.~~ 4. If there is a merge conflict, the setting from the most specific ticket MUST take
 10 precedence [M2.67]. That is, a page-scoped setting in a page-level PrintTicket overwrites
 11 an identical page-scoped setting in a document-level or job-level PrintTicket. Likewise, a
 12 document-scoped setting in a document-level PrintTicket overwrites an identical
 13 document-scoped setting in a job-level PrintTicket.

14 To determine the print settings in the XPS Document the following algorithm should be applied:

- 15 1. Validate the job-level PrintTicket associated with the fixed document sequence by
 16 merging and validating the PrintTicket with the default PrintTicket for the print consumer.
 17 The default PrintTicket represents the default configuration of the print consumer's state;
 18 in the case of a print consumer, the state is the current device and print driver
 19 configuration. Call the resulting ticket the "validated job-level PrintTicket." If no job-level
 20 PrintTicket is supplied, use the default PrintTicket.
- 21 2. For each FixedDocumentX referenced by the fixed document sequence, perform the
 22 following steps:
 - 23 a. Merge and validate the document-level PrintTicket associated with FixedDocumentX
 24 with the validated job-level PrintTicket from Step 1.
 - 25 b. Call the resulting ticket the "validated DocumentX-level PrintTicket." If no document-
 26 level PrintTicket is supplied for the current fixed document, the validated job-level
 27 PrintTicket from Step 1 should be used.
- 28 3. For each FixedPageY referenced by each FixedDocumentX in the fixed document
 29 sequence, perform the following steps:
 - 30 a. Merge and validate the page-level PrintTicket associated with FixedPageY with the
 31 validated DocumentX-level PrintTicket from Step 2.
 - 32 b. Call the resulting ticket the "validated PageXY-level PrintTicket." If no page-level
 33 PrintTicket is supplied for the current fixed page, the validated DocumentX-level
 34 PrintTicket from Step 2 should be used.

35 **9.1.10 SignatureDefinitions Part**

36 Producers MAY add digital signature requests and instructions to an XPS Document in the form
 37 of signature definitions [O2.23]. A producer MAY sign against an existing signature definition to
 38 provide additional signature information [O2.24]. A recipient of the document MAY also sign the
 39 XPS Document against a signature definition [O2.25]. (This is referred to as "co-signing.")

40 Digital signature definitions are stored in a SignatureDefinitions part. A FixedDocument part
 41 refers to a SignatureDefinitions part using a relationship of the SignatureDefinitions type. For
 42 more information, see §H.

43 The SignatureDefinitions part is OPTIONAL [O2.6]. Signature definitions MUST conform to the
 44 Signature Definitions schema as defined in §A [M2.72].

1 For more information on digital signature support in XPS Documents, see §17.

2 **9.1.11 DocumentStructure Part**

3 Explicitly authored document structure information is stored in the DocumentStructure part.
4 This part contains the document outline and defines the framework for every element in fixed
5 pages in terms of semantic blocks, each of which is called a *story*. A story is split into
6 StoryFragments parts, which contain content structure markup that defines semantic blocks
7 such as paragraphs and tables. For more information, see §16.

8 Document structure markup contains a root <DocumentStructure> element. See §16 for
9 markup details. The <DocumentStructure> element uses the Document Structure namespace
10 specified in §H.1.

11 The DocumentStructure part is referenced by relationship from the FixedDocument part, with
12 the relationship type as specified in §H. The content type of the DocumentStructure part is also
13 specified in §H.

14 Consumers MAY provide an algorithmic construction of the structure of an XPS Document based
15 on a page-layout analysis [O2.27], but they MUST NOT use such a method to derive structure
16 for any part of the XPS Document included in the DocumentStructure part [M2.68]. A consumer
17 capable of calculating reading order from the layout of the document MUST use the reading
18 order specified in the DocumentStructure part, even though the derived order might be
19 perceived as preferable to the specified order [M2.68].

20 **9.1.12 StoryFragments Part**

21 The StoryFragments part contains content structure markup (such as tables and paragraphs)
22 associated with a single fixed page.

23 StoryFragments part markup contains a root <StoryFragments> element. See §16 for markup
24 details. The <StoryFragments> element uses the Document Structure namespace specified
25 in §H.1.

26 The StoryFragments part is referenced by relationship from its associated FixedPage part. The
27 content type of the StoryFragments part is specified in §H.2.

9.2 Part Naming Recommendations

Producers and consumers of XPS Documents refer to parts by name and use relationship names to identify the purpose of related parts. The OPC specification describes the syntax for part names. However, following these rules alone can result in a package that is difficult for users to understand. [Example: A user would have to open every Relationship part to know which parts are necessary to accurately render an XPS Document. end example]

By choosing part names according to a well-defined, human-readable convention, the resulting package is easier to browse and specific parts are more easily located. Part names MUST still conform to the syntax specified in the OPC specification [M1.1].

It is RECOMMENDED that producers of XPS Documents use the following part naming convention:

- The FixedDocumentSequence part name SHOULD contain only one segment, and that segment SHOULD have the extension ".fdseq". [Example: "/FixedDocSeq.fdseq" end example] [S2.24].
- A FixedDocument part name SHOULD contain three segments, using "/Documents/n/" in the first two segments and the extension ".fdoc" [S2.25]. Here, *n* SHOULD be a numeral that represents the ordinal position of the fixed document in the fixed document sequence [S2.25]. [Example: The fixed document referenced by the Source attribute of the third <DocumentReference> child of the <FixedDocumentSequence> element could be "/Documents/3/FixedDocument.fdoc". end example]
- A FixedPage part name SHOULD contain four segments, using "/Documents/n/Pages/" as the first three segments and the extension ".fpage" on the last segment [S2.26]. Here, *n* represents the fixed document that includes this page. [Example: The third page of the second document might be "/Documents/2/Pages/3.fpage". end example]
- Resource parts MAY be named to indicate whether their intended use is at the document level or as a shared resource for all documents [O2.28]. A resource that is specific to a particular document SHOULD have a part name that begins with the three segments "/Documents/n/Resources/" where *n* is the particular fixed document [S2.27]. A resource intended to be shared across documents SHOULD begin with the segment "/Resources/" and SHOULD have a final segment that is a globally unique identifier followed by the appropriate extension for that resource [S2.27]. [Example: "/Resources/Fonts/63B51F81-C868-11D0-999C-00C04FD655E1.odttf" end example]

A Font part name SHOULD append the segment "Fonts/" to the resource part name prefix specified above [S2.27]. [Example: A font might be named "/Documents/1/Resources/Fonts/Arial.ttf" or "/Resources/Fonts/F2ABC7B7-C60D-4FB9-AAE4-3CA0F6C7038A.odttf". end example]

An Image part name SHOULD append the segment "Images/" to the resource part name specified above [S2.27]. [Example: An image might be named "/Documents/3/Resources/Images/dog.jpg" or "/Resources/Images/E0D79307-846E-11CE-9641-444553540000.jpg". end example]

A Remote Resource Dictionary part name SHOULD append the segment "Dictionaries/" to the resource part name specified above [S2.27]. Remote resource dictionaries SHOULD also use the ".dict" extension [S2.27]. [Example: A resource dictionary might be named "/Documents/2/Resources/Dictionaries/Shapes.dict" or

- 1 "/Resources/Dictionaries/0DDF3BE2-E692-15D1-AB06-B0AA00BDD685.dict". *end*
- 2 *example*]
- 3 • Any DocumentStructure part name SHOULD contain four segments using
 - 4 "/Documents/*n*/Structure/" as the first three segments and the extension ".struct"
 - 5 [S2.28]. Here *n* represents the fixed document that this structure is associated with.
 - 6 [*Example*: The DocumentStructure part for the first document in a fixed document
 - 7 sequence could be "/Documents/1/Structure/DocStructure.struct". *end example*]
 - 8 • Any StoryFragments part name SHOULD contain five segments using
 - 9 "/Documents/*n*/Structure/Fragments" as the first four segments and the extension
 - 10 ".frag" [S2.29]. Here *n* represents the fixed document that these parts are associated
 - 11 with. [*Example*: A StoryFragment part associated with the third page of the second
 - 12 document in a fixed document sequence could be
 - 13 "/Documents/2/Structure/Fragments/3.frag". *end example*]
 - 14 • ICC profile part names SHOULD contain four segments, using "/Documents/*n*/Metadata/"
 - 15 as the first three segments, where *n* is the fixed document that uses these parts
 - 16 [S2.30]. If an ICC profile part is shared across documents, the part name SHOULD
 - 17 contain two segments, using "/Metadata/" as the first segment and a second segment
 - 18 that is a string representation of a globally unique identifier, followed by an extension
 - 19 [S2.30]. ICC profiles SHOULD use an appropriate extension for the color profile type.
 - 20 [S2.30] [*Example*: ".icm" *end example*]
 - 21 • Thumbnail part names SHOULD contain four segments, using "/Documents/*n*/Metadata/"
 - 22 as the first three segments, where *n* is the fixed document that uses the thumbnail
 - 23 [S2.31]. If the Thumbnail part relates to the package as a whole, the part name
 - 24 SHOULD contain two segments, using "/Metadata/" as the first segment and a second
 - 25 segment that is a string representation of a globally unique identifier, followed by an
 - 26 extension [S2.31]. Thumbnails SHOULD use an extension appropriate to the image type,
 - 27 either ".png" or ".jpg" [S2.31]. [*Example*: A Thumbnail part for a particular fixed page
 - 28 might be "/Documents/1/Metadata/5.png". *end example*]
 - 29 • PrintTicket part names associated with the entire job SHOULD be associated via
 - 30 relationship with the FixedDocumentSequence part and contain two segments, using
 - 31 "/Metadata/" as the first segment [S2.32]. PrintTicket parts associated with a particular
 - 32 fixed document or fixed page SHOULD contain four segments, using
 - 33 "/Documents/*n*/Metadata/" as the first three segments, where *n* is the fixed document
 - 34 that uses these parts [S2.32]. PrintTicket parts SHOULD use the extension ".xml"
 - 35 [S2.32]. [*Example*: A PrintTicket associated with the entire job could be
 - 36 "/Metadata/Job_PT.xml" and a PrintTicket associated with a single page might be
 - 37 "/Documents/1/Metadata/Page2_PT.xml". *end example*]
 - 38 • The names of any non-standard parts that are associated with a particular fixed
 - 39 document SHOULD contain four segments, using "/Documents/*n*/Other/" as the first
 - 40 three segments. Here, *n* is the fixed document to which the part belongs [S2.33].

41 *Example 9-2. XPS Document part naming*

42 An XPS Document that contains two FixedDocument parts is represented as follows:

```
43       /FixedDocSeq.fdseq
44       /Documents/1/FixedDocument.fdoc
45       /Documents/1/Pages/1.fpage
46       /Documents/1/Pages/2.fpage
47       /Documents/1/Resources/Fonts/FontA.ttf
48       /Documents/1/Resources/Images/ImageB.jpg
49       /Documents/1/Metadata/Document_PT.xml
```

```
1 /Documents/1/Metadata/Page5_PT.xml
2 /Documents/1/Structure/DocStructure.struct
3 /Documents/1/Structure/Fragments/1.frag
4 /Documents/1/Structure/Fragments/2.frag
5 /Documents/1/Other/FabrikamIncBussinessAccount.xml
6 /Documents/2/FixedDocument.fdoc
7 /Documents/2/Pages/1.fpage
8 /Documents/2/Resources/Fonts/FontB.ttf
9 /Documents/2/Resources/Images/ImageA.png
10 /Documents/2/Metadata/ColorProfile.icm
11 /Documents/2/Metadata/Document_PT.xml
12 /Documents/2/Other/FabrikamIncInsuranceInfo.xml
13 /Metadata/Job_PT.xml
14 /Resources/Fonts/63B51F81-C868-11D0-999C-00C04FD655E1.ttf

15 end example]
```

16 9.3 XPS Document Markup

17 XPS Document markup has been designed to facilitate the independent development of
18 compatible systems that produce or consume XPS Documents. It also shares concepts with
19 portions of the Microsoft .NET Framework 3.0 programming platform.

20 The graphics rendering model is shared with that of the Windows Presentation Foundation,
21 assuring fidelity between on-screen display and printed output. The syntax of fixed page, fixed
22 document, and fixed document sequence markup is compatible with that of Windows
23 Presentation Foundation XAML. The elements, attributes, and attribute values are a subset of
24 those defined by the Windows Presentation Foundation.

25 The relationship between XPS Document markup and .NET 3.0 technologies does not impose
26 any requirement on system implementations. Support for XPS Document markup does not
27 require incorporation of .NET 3.0, the Windows Presentation Foundation, or managed code.
28 However, the relationship with .NET 3.0 technologies allows producers to extend XPS Document
29 markup for further use in the Windows Presentation Foundation framework, such as by
30 including additional presentation features.

31 The design of XPS Document markup reflects the tradeoffs between two, sometimes competing,
32 goals:

- 33 1. XPS Document markup should be parsimonious; that is, it should include only the
34 minimum set of primitive operations and markup constructs necessary to render text and
35 graphics with full fidelity. Redundancy in the specification increases the opportunity for
36 independent implementations, such as printer-resident raster image processors (RIPs),
37 viewers, and interactive applications, to introduce accidental incompatibilities.
38 Redundancy also increases the cost of implementation and testing, and, typically, the
39 required memory footprint.
- 40 2. XPS Document markup should be compact; that is, the most common graphical
41 primitives for vector graphics and text-rendering should have compact representations.
42 Bloated representations compromise the performance of systems handling XPS
43 Documents. As byte-count increases, so does communication time. Although
44 compression can be used to improve communication time, it cannot eliminate the
45 performance loss caused by bloated representations.

9.3.1 Support for Versioning and Extensibility

XPS Document markup has been designed in anticipation of the evolution of this specification. It also allows third parties to extend the markup. XPS Document markup incorporates the Markup Compatibility and Extensibility specification incorporated by the Office Open XML specification.

The following parts MAY include elements and attributes defined in the Markup Compatibility and Extensibility specification [O2.29]:

- DocumentStructure
- FixedDocument
- FixedDocumentSequence
- FixedPage
- Relationships
- Remote Resource Dictionary
- SignatureDefinitions
- StoryFragments

Consumers of these parts MUST support the Markup Compatibility and Extensibility specification [M2.69]. Before attempting to validate one of these parts against a schema, processors MUST remove all markup compatibility elements and attributes and all ignorable elements and attributes not defined in the expected version of XPS Document markup [M2.69].

Markup compatibility elements and attributes that appear in one XPS Document part do not carry through to a second part via an inline URI reference in the XML markup. Likewise the markup compatibility mechanisms do not carry through from part to part via relationship.

9.3.2 XML Usage

All XML content of the parts defined in this specification MUST conform to the following validation rules:

1. XML content MUST be encoded using either UTF-8 or UTF-16. If any such part includes an encoding declaration (as defined in §4.3.3 of the XML specification), that declaration MUST NOT name any encoding other than UTF-8 or UTF-16 [M2.70].
2. The XML 1.0 specification allows for the usage of Data Type Definitions (DTDs), which enable Denial of Service attacks, typically through the use of an internal entity expansion technique. As mitigation for this potential threat, DTD content MUST NOT be used in the XML markup defined in this specification, and consumers MUST treat the presence of DTD content as an error [M2.71].
3. If the XML content contains the Markup Compatibility and Extensibility namespace, as described in the Markup Compatibility and Extensibility specification, it MUST be processed to remove Markup Compatibility and Extensibility elements and attributes, ignorable namespace declarations, and ignored elements and attributes before applying further validation rules below [M2.69].
4. XML content MUST be valid against the corresponding W3C XSD schema defined in this specification. In particular, the XML content MUST NOT contain elements or attributes drawn from namespaces that are not explicitly defined in the corresponding XSD unless

1 the XSD allows elements or attributes drawn from any namespace to be present in
2 particular locations in the XML markup [M2.72].

3 5. XML content MUST NOT contain elements or attributes drawn from "xml" or "xsi"
4 namespaces unless they are explicitly defined in the W3C XSD schema or by other means
5 in the specification [M2.73].

6 **9.3.3 Markup Model**

7 XPS Document markup is an XML-based markup language that uses elements, attributes, and
8 namespaces. The schema for XPS Document markup includes only elements and their
9 attributes, comments, and whitespace. Arbitrary character data intermingled in the markup is
10 not allowed.

11 Fixed page markup is expressed using elements and attributes and is based on a higher-level
12 abstract model of contents and properties. Some fixed page elements can hold "contents,"
13 which are expressed as child elements. Properties can be expressed either as attributes or child
14 elements.

15 XPS Document markup also uses resources and resource dictionaries, which allow elements to
16 share property values.

17 **9.3.3.1 Namespaces**

18 The following XML namespaces are defined for use in XPS Document markup:

- 19 • The XPS Document namespace, the principal namespace used for elements and
20 attributes in fixed page markup. For more information, see §H.
- 21 • The Resource Dictionary Key namespace, which allows certain XPS Document elements
22 to be included in a resource dictionary, as described in §14.2.
- 23 • The Markup Compatibility namespace, which supports the Markup Compatibility and
24 Extensibility specification as defined in the OPC specification.

25 **9.3.3.2 Properties**

26 A *property* is a characteristic of an element. XPS Document property values can be expressed
27 either as property attributes or property elements. *Property values* can be stored in a resource
28 dictionary and referenced by an attribute that uses a special syntax to express its value. For
29 more information, see §14.2.

30 Properties MUST NOT be set more than once, regardless of the syntax used to specify the value
31 [M2.74]. In certain cases, they can be specified using either property attributes or property
32 elements. Consumers MUST treat properties that are specified in both ways as an error
33 [M2.74].

34 Some properties are common to several fixed page elements. For more information, see §14.

35 **9.3.3.2.1 Composable Property Values**

36 Some fixed page properties are composable, meaning that the page marking effect is
37 determined by combining the property value of a given element with that of its parent and
38 ancestor elements. [*Example: A <Path> element with an Opacity value of 0.5 nested inside a
39 <Canvas> element with an Opacity value of 0.5 results in an effective 25% opacity of the
40 <Path> element when rendered. end example]*]

1 The coordinate space used to render page marking elements is also composable. By default,
2 elements are rendered in a coordinate space with units of 1/96". The *effective coordinate space*
3 for a particular element is created by sequentially applying each parent and ancestor element's
4 affine matrix transformation, specified with the Transform or RenderTransform properties, from
5 outermost to innermost, including the element's own affine matrix transformation.

6 For more information, see §18.1.3, and §18.5.

7 **9.3.3.2.2 Property Attribute Syntax**

8 Some property values can be expressed using simple XML attribute syntax, that is, with a text
9 string. The value of properties used to describe geometries can be expressed using an
10 abbreviated syntax. For more information, see §11.2.3.

11 *Example 9-3. Property attribute syntax*

12 The following syntax can be used to specify the color of a brush:

```
13 <!-- Property Attribute Syntax -->  
14 <SolidColorBrush Color="#FF0000" />
```

15 *end example]*

16 **9.3.3.2.3 Property Element Syntax**

17 Some property values can also be expressed using a child element to describe the property
18 value. These property elements are included to enable usage of the markup compatibility
19 mechanisms described in the Markup Compatibility and Extensibility specification. The element
20 name is derived from a combination of a parent element name and the property name,
21 separated by a dot (.) character.

22 The order of child property elements is significant: they **MUST** occur before any contents of the
23 parent element and they **MUST** appear in the sequence specified in the schema [M2.72].

1 *Example 9–4. Property element syntax*

2 When specifying Clip and RenderTransform properties of the canvas, both must appear before
3 any path and glyphs contents of the canvas.

```

4     <Canvas>
5         <!-- First, the property-related child elements -->
6         <Canvas.RenderTransform>
7             <MatrixTransform Matrix="1,0,0,1,0,0" />
8         </Canvas.RenderTransform>
9         <Canvas.Clip>
10            <PathGeometry>
11                ...
12            </PathGeometry>
13        </Canvas.Clip>
14        <!-- Then, the "contents" -->
15        <Path ...>
16            ...
17        </Path>
18        <Glyphs ... />
19    </Canvas>

```

20 *end example]*

21 **9.3.4 Whitespace**

22 XPS Documents allow flexible whitespace usage in markup. Wherever a single whitespace
23 character is allowed, multiple whitespace characters MAY be used [O2.30]. Attributes that
24 specify comma-delimited attribute values MAY, unless specified otherwise, OPTIONALLY include
25 whitespace characters preceding or following the comma [O2.31]. XPS Document markup MUST
26 NOT use the xml:space attribute [M2.75]. Additionally, where the XPS Document schema
27 specifies attributes of types that allow whitespace collapsing, leading and trailing whitespace in
28 the attribute value MAY be used along with other whitespace that relies on the whitespace
29 collapsing behavior specified in the XML Schema Specification [O2.32].

30 [*Note: Consult the XPS Document Schema for exact whitespace allowed. end note]*

31 **9.3.5 Language**

32 Language information supports the following features:

- 33 • Language-dependent find features
- 34 • Selection of a text-to-speech dictionary by a screen-reading program (to provide
35 accessibility to persons with disabilities)
- 36 • Selection of a spelling checker for text copied to another document
- 37 • Selection of a grammar checker for text copied to another document
- 38 • Correct font rendering when copying the text to another document

39 The last point refers to instances in which multiple languages share the same script. [*Example:*
40 The Devanagari script is shared by the Indic languages Bhojpuri, Bihari, Hindi, Kashmiri,
41 Konkani, Marathi, Nepali, and Sanskrit. However, these languages render certain glyph
42 sequences differently. When text is copied from an XPS Document, the language of the copied
43 characters is needed to ensure proper rendering of the glyphs when they are pasted into

1 another application. This scenario applies to most Indic-language fonts, some East Asian-
2 language fonts, and others. *end example*]

3 **9.3.5.1 xml:lang Attribute**

4 The language of the contents of an XPS Document MUST be identified using the xml:lang
5 attribute, the value of which is inherited by child and descendant elements [M2.76]. This
6 attribute is defined in the W3C XML specification.

7 xml:lang is REQUIRED for <FixedPage> elements and MAY be used with <Canvas>, <Path>, and
8 <Glyphs> elements; it is not valid on any other fixed page markup element [M2.72]. xml:lang is
9 also REQUIRED for the <DocumentOutline> element for document structure and OPTIONAL for
10 the <OutlineEntry> element [M2.72]. When the language of the contents is unknown and is
11 required, the value "und" (undetermined) MUST be used [M2.76].

10. Documents

XPS Documents contain a root fixed document sequence that binds a collection of fixed documents which, in turn, bind a collection of fixed pages. All page markings are specified with <Glyphs> or <Path> elements on the fixed page. These elements can be grouped within one or more <Canvas> elements. Page markings are positioned by real-number coordinates in the coordinate space of the fixed page. The coordinate space can be altered by applying a render transformation.

10.1 <FixedDocumentSequence> Element

element **FixedDocumentSequence**

diagram	
annotation	Specifies a sequence of fixed documents.

The <FixedDocumentSequence> element contains one or more <DocumentReference> elements. The order of <DocumentReference> elements MUST match the order of the documents in the fixed document sequence [M3.1].

Example 10-1. <FixedDocumentSequence> usage

```

<FixedDocumentSequence xmlns="http://schemas.microsoft.com/xps/2005/06">
  <DocumentReference Source="Documents/1/FixedDocument.fdoc" />
  <DocumentReference Source="Documents/2/FixedDocument.fdoc" />
</FixedDocumentSequence>
    
```

end example]

10.1.1 <DocumentReference> Element

element **DocumentReference**

diagram													
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Source</td> <td>xs:anyURI</td> <td>required</td> <td></td> <td></td> <td>Specifies the URI of the fixed document content. The specified URI MUST refer to a FixedDocument part within the XPS Document [M3.2].</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Source	xs:anyURI	required			Specifies the URI of the fixed document content. The specified URI MUST refer to a FixedDocument part within the XPS Document [M3.2].
Name	Type	Use	Default	Fixed	Annotation								
Source	xs:anyURI	required			Specifies the URI of the fixed document content. The specified URI MUST refer to a FixedDocument part within the XPS Document [M3.2].								

annotation	Contains a reference to a FixedDocument part.
------------	---

1 The <DocumentReference> element specifies a FixedDocument part as a URI in the Source
 2 attribute. Producers MUST NOT produce a document with multiple <DocumentReference>
 3 elements that reference the same fixed document [M3.3].

4 **10.2 <FixedDocument> Element**

5 element **FixedDocument**

diagram	
annotation	Binds an ordered sequence of fixed pages together into a single multi-page document.

6 The <FixedDocument> element contains one or more <PageContent> elements. The order of
 7 <PageContent> elements MUST match the order of the pages in the document [M3.4].

8 *Example 10-2. <FixedDocument> usage*

```

9     <FixedDocument xmlns="http://schemas.microsoft.com/xps/2005/06">
10       <PageContent Source="Pages/1.fpage" />
11       <PageContent Source="Pages/2.fpage" />
12     </FixedDocument>
    
```

13 *end example]*

14 **10.2.1 <PageContent> Element**

15 element **PageContent**

diagram													
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Source</td> <td>xs:anyURI</td> <td>required</td> <td></td> <td></td> <td>Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Source	xs:anyURI	required			Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].
Name	Type	Use	Default	Fixed	Annotation								
Source	xs:anyURI	required			Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].								

	Width	<u>ST_GEOne</u>				Typical width of pages contained in the page content.
	Height	<u>ST_GEOne</u>				Typical height of pages contained in the page content.
annotation	Defines a reference from a fixed document to a part that contains a <FixedPage> element.					

1 Each <PageContent> element refers to the source of the content for a single page. The number
2 of pages in the document can be determined by counting the number of <PageContent>
3 elements.

4 The <PageContent> element has a single required attribute, Source, which refers to a
5 FixedPage part. It can optionally include advisory Height and Width attributes to indicate the size
6 of a single page. (The authoritative height and width are specified by the fixed page.) The
7 Height and Width attribute values allow consumers such as viewers to make initial visual layout
8 estimates quickly, without loading and parsing all of the individual fixed pages. These
9 consumers then update the page dimensions when the fixed page is loaded, if they differ.

10 The <PageContent> element has one allowable child element, <PageContent.LinkTargets>, and
11 it MUST NOT contain more than a single child element [M2.72].

12 Producers MUST NOT produce markup where a <PageContent> element references the same
13 fixed page referenced by any other <PageContent> element in the entire XPS Document, even
14 in other fixed documents within the fixed payload [M3.6].

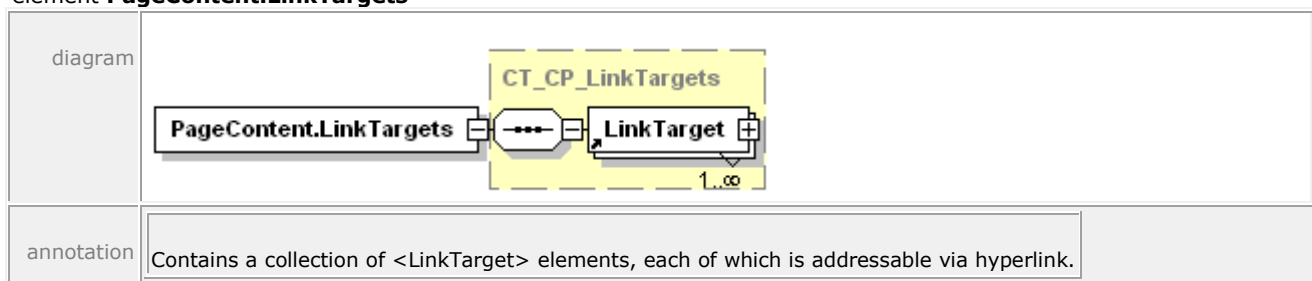
15 *Example 10-3. <PageContent> usage*

```
16 <FixedDocument xmlns="http://schemas.microsoft.com/xps/2005/06">
17 <PageContent Source="Pages/1.fpage" Height="1056" Width="816" />
18 <PageContent Source="Pages/2.fpage" Height="1056" Width="816" />
19 </FixedDocument>
```

20 *end example]*

21 10.2.2 <PageContent.LinkTargets> Element

22 element **PageContent.LinkTargets**



23 The <PageContent.LinkTargets> element defines the list of link targets that specify each named
24 element on the page that can be addressed by hyperlink.

1 *Example 10-4. <PageContent.LinkTargets> usage*

2 In the following markup, `Pages/2.fpage` contains two `<LinkTarget>` elements with Name
 3 attribute values of `Anchor1` and `Anchor2`:

```

4     <FixedDocument xmlns="http://schemas.microsoft.com/xps/2005/06">
5         <PageContent Source="Pages/1.fpage" Height="1056" Width="816" />
6         <PageContent Source="Pages/2.fpage" Height="1056" Width="816">
7             <PageContent.LinkTargets>
8                 <LinkTarget Name="Anchor1" />
9                 <LinkTarget Name="Anchor2" />
10            </PageContent.LinkTargets>
11        </PageContent>
12    </FixedDocument>
    
```

13 *end example]*

14 **10.2.3 <LinkTarget> Element**

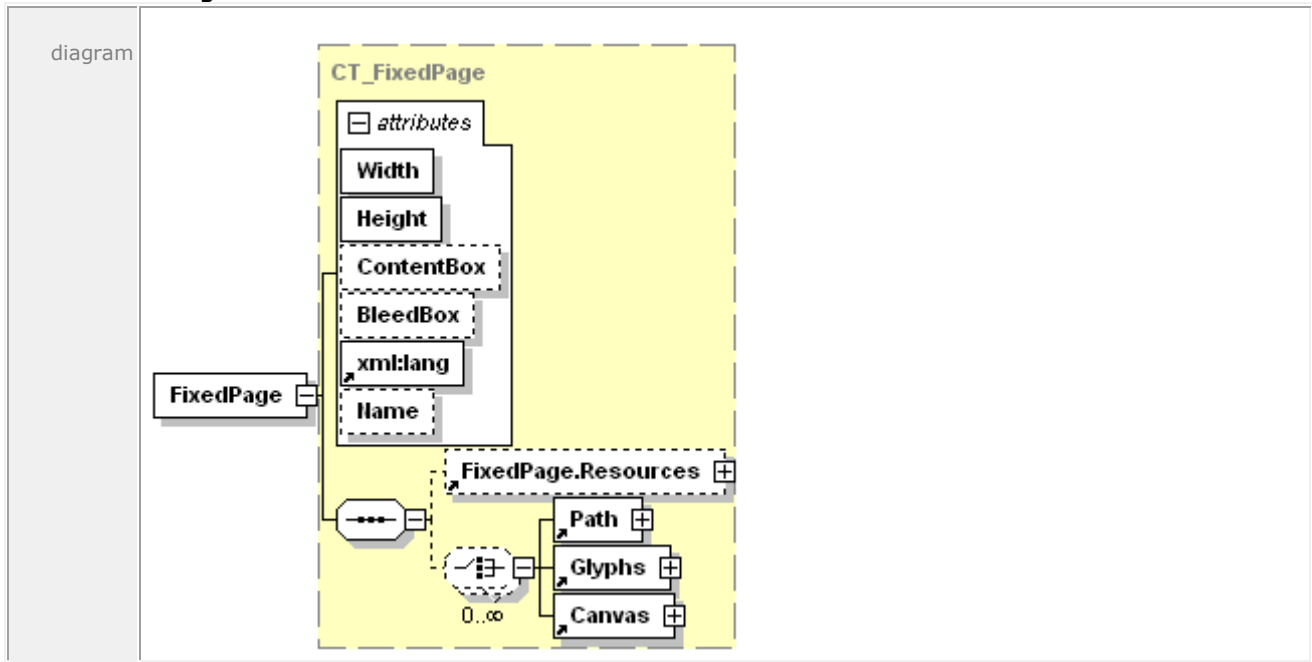
15 element **LinkTarget**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Name	<u>ST_Name</u>	required			Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
annotation	Specifies an addressable point on the page.					

16 The `<LinkTarget>` element specifies a Name attribute, which corresponds to a named location
 17 within the fixed page specified by its parent `<PageContent>` element. By encapsulating this
 18 information in the fixed document, consumers do not need to load every `FixedPage` part to
 19 determine if a particular Name value exists in the document. For more information, see §16.2.

1 **10.3 <FixedPage> Element**

2 element **FixedPage**



Name	Type	Use	Default	Fixed	Annotation
Width	<u>ST_GEOne</u>	required			Width of the page, expressed as a real number in units of the effective coordinate space.
Height	<u>ST_GEOne</u>	required			Height of the page, expressed as a real number in units of the effective coordinate space.
ContentBox	<u>ST_ContentBox</u>				Specifies the area of the page containing imageable content that is to be fit within the imageable area when printing or viewing. Contains a list of four coordinate values (ContentOriginX, ContentOriginY, ContentWidth, ContentHeight), expressed as comma-separated real numbers. Specifying a value is RECOMMENDED [S3.1]. If omitted, the default value is (0,0,Width,Height).
BleedBox	<u>ST_BleedBox</u>				Specifies the area including crop marks that extends outside of the physical page. Contains a list of four coordinate values (BleedOriginX, BleedOriginY, BleedWidth, BleedHeight), expressed as comma-separated real numbers. If omitted, the default value is (0,0,Width,Height).
xml:lang		required			Specifies the default language used for the current element and for any child or descendant elements.

						The language is specified according to RFC 3066.
	Name	<u>ST_Name</u>				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
annotation	Contains markup that describes the rendering of a single page of content.					

1 The <FixedPage> element contains the contents of a page and is the root element of a
 2 FixedPage part. The fixed page contains the elements that together form the basis for all
 3 markings rendered on the page: <Paths>, <Glyphs>, and the optional <Canvas> grouping
 4 element.

5 The fixed page MUST specify a height, width, and default language [M2.72].

6 The coordinate space of the fixed page is composable, meaning that the marking effects of its
 7 child and descendant elements are affected by the coordinate space of the fixed page.

8 *Example 10-5. Fixed page markup*

```

9     <FixedPage Height="1056" Width="816" xml:lang="en-US"
10       xmlns="http://schemas.microsoft.com/xps/2005/06">
11       <Glyphs
12         OriginX="96"
13         OriginY="96"
14         UnicodeString="This is Page 1!"
15         FontUri="../Resources/Fonts/Times.TTF"
16         FontRenderingEmSize="16" />
17     </FixedPage>

```

18 *end example]*

19 **10.3.1 BleedBox Attribute**

20 The BleedBox attribute defines the area (inclusive of crop marks) that extends outside of the
 21 physical page. The bleed box is expressed as four comma-separated, real-number coordinate
 22 values that correspond to BleedOriginX, BleedOriginY, BleedWidth, BleedHeight. These values
 23 are specified in units of 1/96".

24 Bleed boxes that do not satisfy the following conditions are invalid and SHOULD be ignored in
 25 favor of the default bleed box [S3.2]:

- 26 • The BleedBox BleedOriginX value MUST be less than or equal to 0 [M3.7].
- 27 • The BleedBox BleedOriginY value MUST be less than or equal to 0 [M3.8].
- 28 • The BleedBox BleedWidth value MUST be greater than or equal to the fixed page Width
 29 attribute value plus the absolute value of the Bleedbox BleedOriginX value [M3.9].
- 30 • The BleedBox BleedHeight value MUST be greater than or equal to the fixed page Height
 31 attribute value plus the absolute value of the BleedBox BleedOriginY value [M3.10].

32 If the BleedBox attribute is omitted, the default value is "0,0,Width,Height".

10.3.2 ContentBox Attribute

The ContentBox attribute specifies the area of the page that contains imageable content that must fit in the imageable area when printing or viewing. Specifying this attribute is RECOMMENDED [S3.1]. The content box is expressed as four comma-separated, real-number coordinate values that correspond to ContentOriginX, ContentOriginY, ContentWidth, ContentHeight. These values are specified in units of 1/96".

Content boxes that do not satisfy the following conditions are invalid and SHOULD be ignored in favor of the default content box [S3.3]:

- The ContentBox ContentOriginX value MUST be greater than or equal to 0 and less than the fixed page Width attribute value [M3.11].
- The ContentBox ContentOriginY value MUST be greater than or equal to 0 and less than the fixed page Height attribute value [M3.12].
- The ContentBox ContentWidth value MUST be less than or equal to the difference between the fixed page Width attribute value and the ContentBox ContentOriginX value [M3.13].
- The ContentBox ContentHeight value MUST be less than or equal to the difference between the fixed page Height attribute value and the ContentBox ContentOriginY value [M3.14].

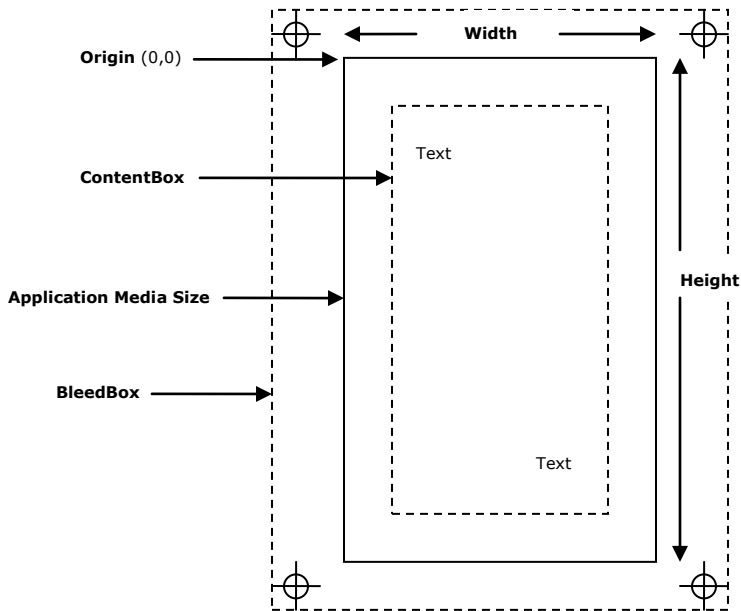
If the ContentBox attribute is omitted, the default value is "0,0,Width,Height".

10.3.3 Page Size Terminology

The following terminology is used to describe page sizes:

- The *producer media size* represents the media size that is used by the producer for layout purposes. This size is described by the Height and Width attributes.
- The *producer bleed size* represents the overflow or "bleed" box used by the producer for registration and layout. This size is described by the BleedBox attribute. If the BleedBox attribute is not present, then the producer bleed size is defined as the producer media size.
- The *producer content size* represents the content bounding box specified by the producer. This size is described by the ContentBox attribute. If the ContentBox attribute is not present, then the producer content size is defined as the producer media size.
- The *physical media size* represents the physical media on which the content will be printed. This size is described by the PageMediaSize keyword in the Print Schema. The PageMediaSize is page-orientation-agnostic. This means that the representation of physical media size is defined by two dimensions, PageMediaSizeWidth and PageMediaSizeHeight. For more information, see the Print Schema specification.
- The *physical imageable size* represents the area that is printable by a specific device. This size is described by the PageImageableSize keyword in the Print Schema. The PageImageableSize is relative to the physical media size (PageMediaSize keyword) and the orientation (PageOrientation keyword). For more information, see the Print Schema specification.

1 *Figure 10-1. Page regions*



2

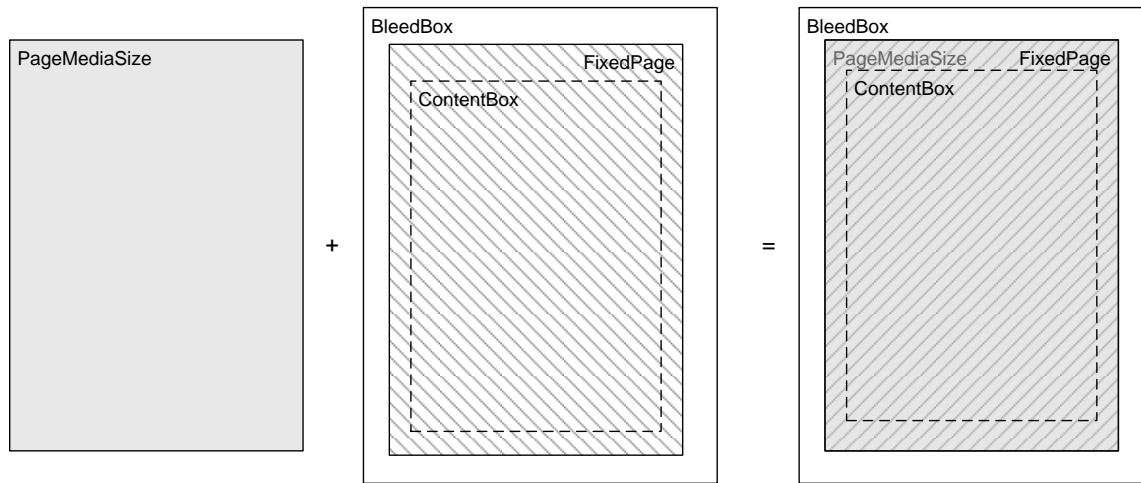
3 **10.3.4 Media Orientation and Scaling**

4 When rendering a fixed page for printing, consumers **MUST** be aware of the interaction between
 5 the fixed page markup and the PrintTicket settings [M3.15]. The interaction for media scaling is
 6 governed by the PageMediaSize, PageImageableSize, PageScaling, and PageOrientation Print
 7 Schema keywords.

8 For media orientation, the PageMediaSize, PageOrientation and the width and height of the
 9 <FixedPage> element determine the rendering of the fixed page. In the absence of media
 10 scaling, the fixed page content is imaged directly to the physical media with the origin of the
 11 fixed page aligned with the origin of the physical media size. Any fixed page content that
 12 extends beyond the dimension of the physical media size **SHOULD** be clipped [S3.4].

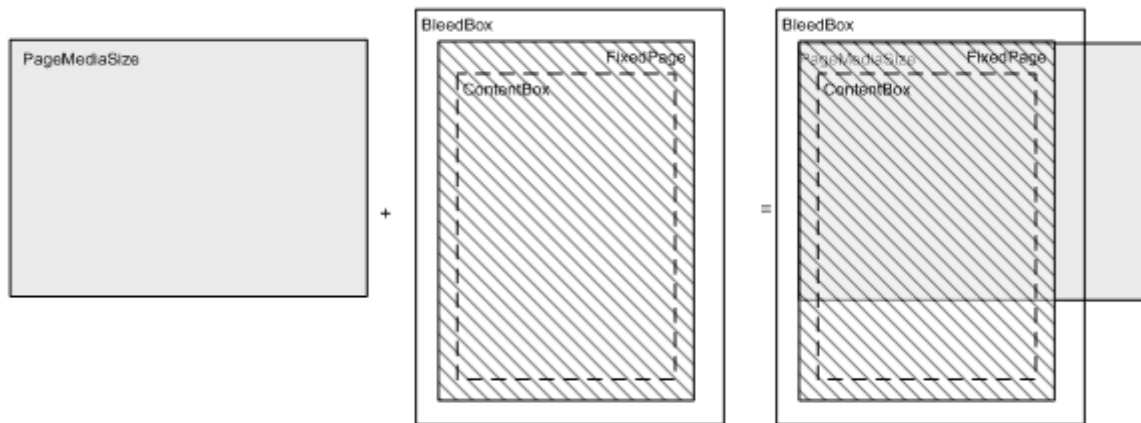
13 *[Example: Consider a fixed page with a width of 816 (8.5") and a height of 1056 (11"). If the*
 14 *PrintTicket specifies the PageMediaSize value as Letter and the PageOrientation value as*
 15 *Portrait then no clipping occurs, as shown below.*

1 *Figure 10–2. Matching PrintTicket and fixed page size and orientation*



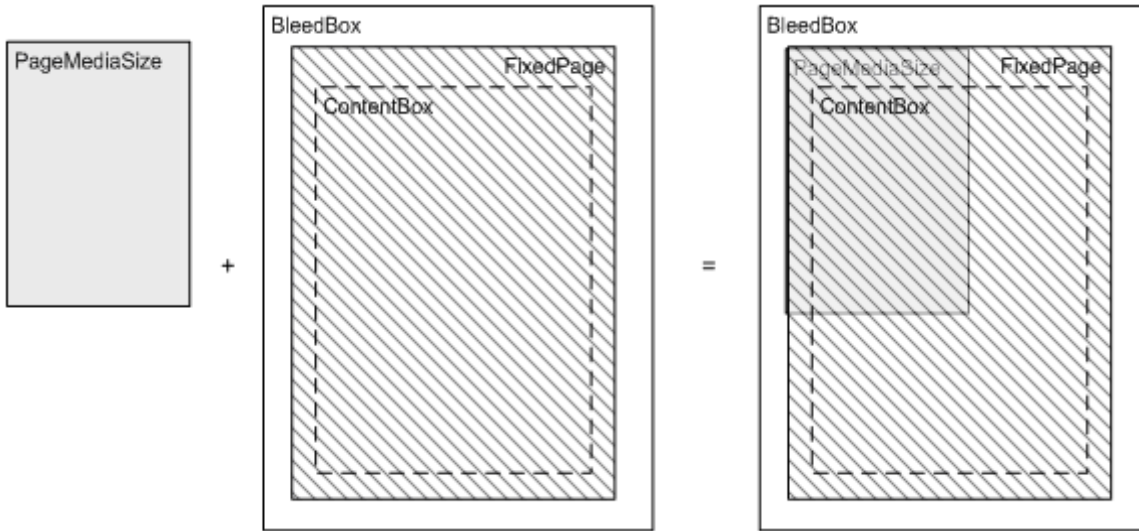
2
 3 If the PrintTicket specifies the PageMediaSize value as Letter and the PageOrientation value as
 4 Landscape the bottom of the fixed page content is clipped, as shown below.

5 *Figure 10–3. Matching PrintTicket and fixed page size with differing orientation*



6
 7 Finally, if the PrintTicket specifies the PageMediaSize value as 4x6 and the PageOrientation
 8 value as Portrait, the bottom and right portions of the fixed page content are clipped, as shown
 9 below.

1 *Figure 10–4. Matching PrintTicket and fixed page orientation with differing size*



2
3 *end example]*

4 [Note: Fixed pages intended for landscape-orientation printing must be produced differently
5 than those intended for portrait-orientation printing; their Width attribute holds the larger value,
6 their Height attribute holds the smaller value.

7 The PageOrientation setting in the PrintTicket does not rotate the fixed page, but instead
8 determines how the PageMediaSize dimensions relate to the PageImageableSize dimensions
9 reported by the device.

10 As a consequence, in order to print a fixed page that has been produced with a landscape
11 orientation, either the PrintTicket would specify PageMediaSize and PageOrientation values such
12 that the dimensions of the fixed page match the PageImageableSize reported by the device, or
13 it would specify an appropriate PageScaling option. *end note]*

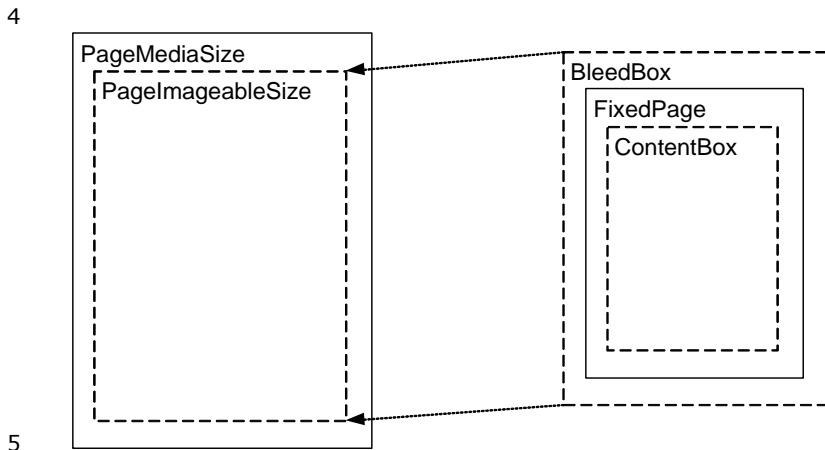
14 For media scaling, the following page scaling options determine the rendering of the fixed page:

- 15 • FitApplicationBleedSizeToPageImageableSize
- 16 • FitApplicationContentSizeToPageImageableSize
- 17 • FitApplicationMediaSizeToPageImageableSize
- 18 • FitApplicationMediaSizeToPageMediaSize

19 This is not an exhaustive list; for more information, see the Print Schema specification.

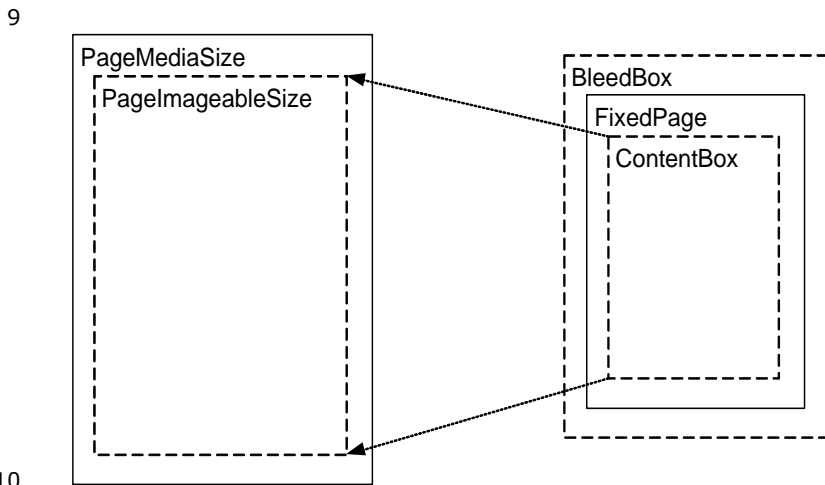
1 **10.3.4.1 FitApplicationBleedSizeToPageImageableSize**

2 Consumers MUST scale the bleed box (producer bleed size) to the PageImageableSize,
 3 preserving the aspect ratio [M3.16]. See the Print Schema PageScaling definition.



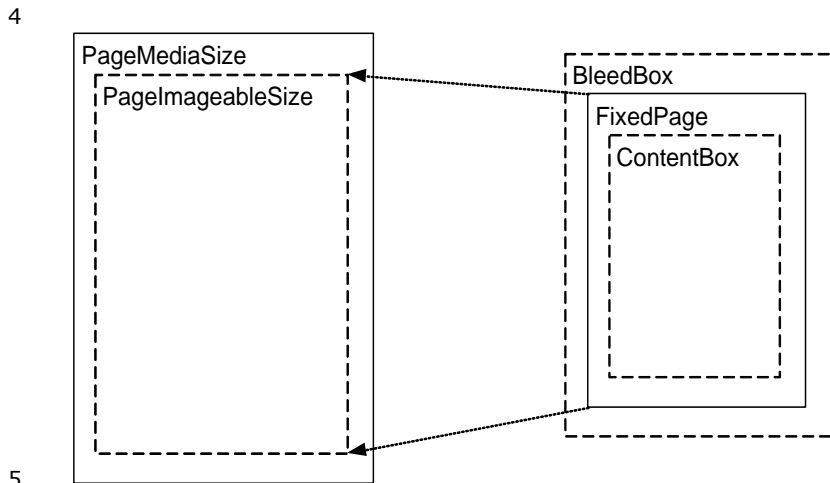
6 **10.3.4.2 FitApplicationContentSizeToPageImageableSize**

7 Consumers MUST scale the content box (producer content size) to the PageImageableSize,
 8 preserving the aspect ratio [M3.17]. See the Print Schema PageScaling definition.



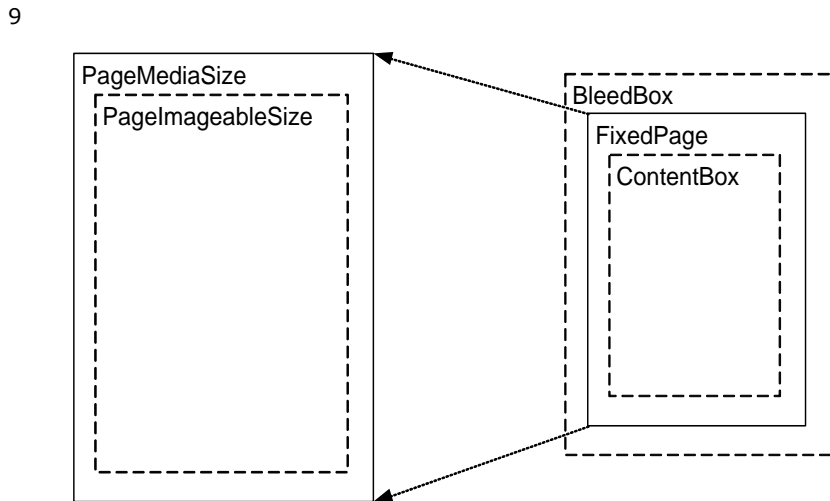
1 **10.3.4.3 FitApplicationMediaSizeToPageImageableSize**

2 Consumers MUST scale the height and width (producer media size) to the PageImageableSize,
3 preserving the aspect ratio [M3.18]. See the Print Schema PageScaling definition.



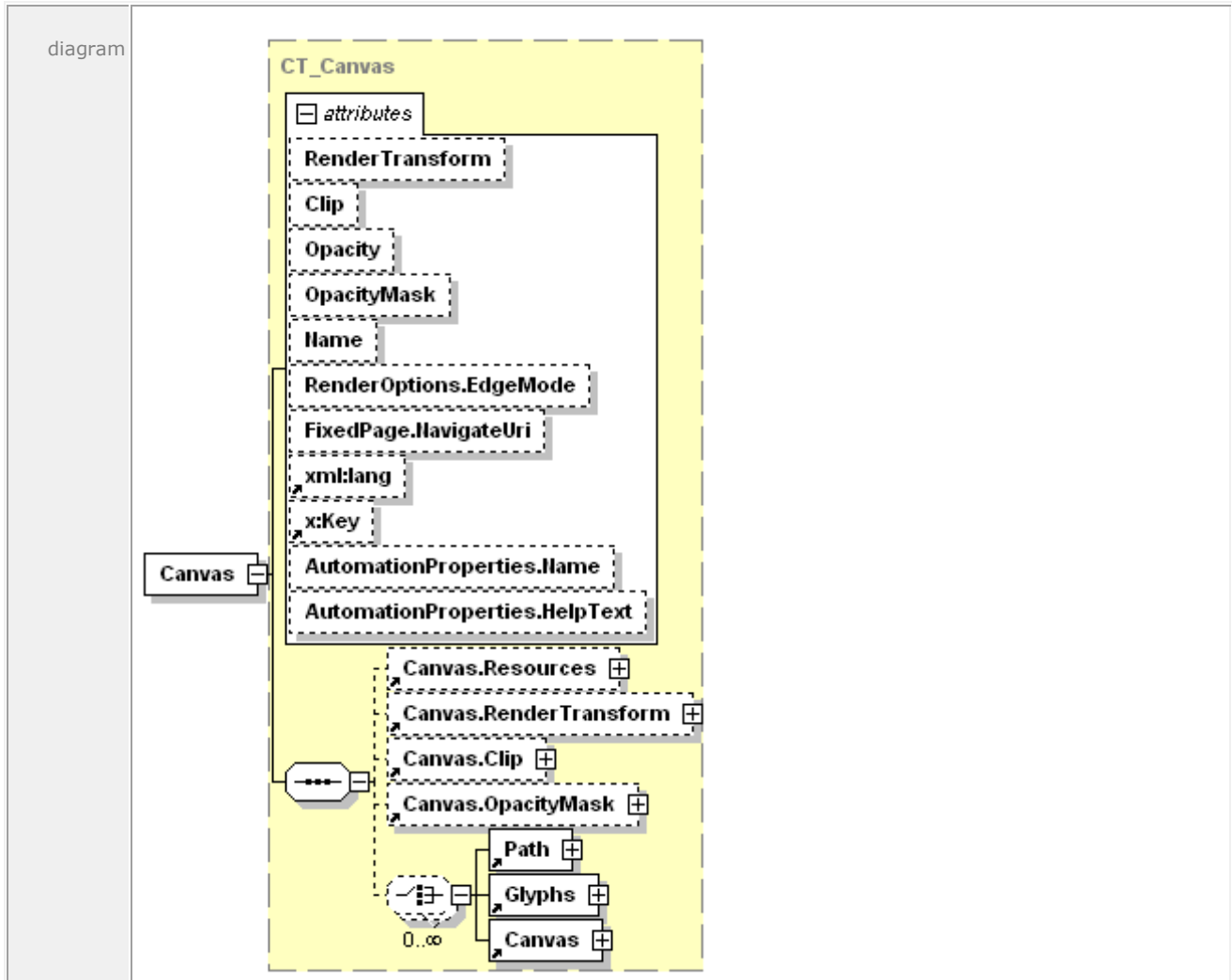
6 **10.3.4.4 FitApplicationMediaSizeToPageMediaSize**

7 Consumers MUST scale the height and width (producer media size) to the PageMediaSize,
8 preserving the aspect ratio [M3.19]. See the Print Schema PageScaling definition.



1 **10.4 <Canvas> Element**

2 element **Canvas**



attributes	Name	Type	Use	Default	Fixed	Annotation
	RenderTransform	ST_RscRefMatrix				Establishes a new coordinate frame for the child and descendant elements of the canvas, such as another canvas. Also affects clip and opacity mask.
	Clip	ST_RscRefAbbrGeomF				Limits the rendered region of the element.
	Opacity	ST_ZeroOne		1.0		Defines the uniform

					transparency of the canvas. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
OpacityMask	<u>ST_RscRef</u>				Specifies a mask of alpha values that is applied to the canvas in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
Name	<u>ST_Name</u>				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
RenderOptions.EdgeMode	<u>ST_EdgeMode</u>				Controls how edges of paths within the canvas are rendered. The only valid value is Aliased. Omitting this attribute causes the edges to be rendered in the consumer's default manner.
FixedPage.NavigateUri	xs:anyURI				Associates a hyperlink URI with the element. May be a relative reference or a URI that addresses a resource that is internal to or external to the package.
xml:lang					Specifies the default language used for the current element and for any child or descendant elements. The language is specified according to RFC 3066.
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M3.20].

	AutomationProperties.Name	xs:string				A brief description of the <Canvas> contents for accessibility purposes, particularly if filled with a set of vector graphics and text elements intended to comprise a single vector graphic.
	AutomationProperties.HelpText	xs:string				A detailed description of the <Canvas> contents for accessibility purposes, particularly if filled with a set of graphics and text elements intended to comprise a single vector graphic.
annotation	Groups <FixedPage> descendant elements together.					

1 The <Canvas> element groups elements together. [*Example: <Glyphs> and <Path> elements*
2 can be grouped in a canvas in order to be identified as a unit (as a hyperlink destination) or to
3 apply a composed property value to each child and ancestor element. *end example*]

4 Some properties of the <Canvas> element are composable and affect the rendering of child
5 elements. This includes the coordinate space of the canvas. For details, see §14.

6 The RenderOptions.EdgeMode property can be set on the <Canvas> element to instruct anti-
7 aliasing consumers to render the contents of the <Canvas> and all child and descendant
8 elements without performing anti-aliasing, including child brushes and their contents as well as
9 contents included via resource dictionary references.

10 *Example 10–6. Canvas composition*

11 The following markup describes a path that provides the background. On top of this is rendered
12 a canvas with the composable Opacity and RenderTransform properties specified.

13 The path inside the canvas has the same path geometry as the background path, but since it is
14 composing the <Canvas> element's RenderTransform property, it is rendered differently. The
15 path is partially transparent due to the composable Opacity property of the parent <Canvas>
16 element.

17 The <Glyphs> element inside the canvas specifies its own RenderTransform property. This
18 property is composed with the <Canvas> element's RenderTransform property, such that the
19 coordinate space of the <Glyphs> element is transformed within the context of the coordinate
20 space transformed by the <Canvas> element.

```

21 <Path>
22   <Path.Fill>
23     <SolidColorBrush Color="#808080" />
24   </Path.Fill>
25   <Path.Data>
26     <PathGeometry>
27       <PathFigure StartPoint="0,0" IsClosed="true">
28         <PolyLineSegment Points="200,0 200,100 0,100 0,0" />

```

```

1         </PathFigure>
2     </PathGeometry>
3 </Path.Data>
4 </Path>
5
6 <Canvas Opacity="0.5" RenderTransform="0.75,0,0,0.75,25,46">
7     <Path>
8         <Path.Fill>
9             <SolidColorBrush Color="#0000FF" />
10        </Path.Fill>
11        <Path.Data>
12            <PathGeometry>
13                <PathFigure StartPoint="0,0" IsClosed="true">
14                    <PolyLineSegment Points="200,0 200,100 0,100 0,0" />
15                </PathFigure>
16            </PathGeometry>
17        </Path.Data>
18    </Path>
19    <Glyphs
20        FontUri=" ../Resources/Fonts/times.ttf"
21        OriginX="1"
22        OriginY="100"
23        UnicodeString="EXAMPLE"
24        FontRenderingEmSize="42"
25        RenderTransform="1.0,0,0,2.0,0,-100">
26        <Glyphs.Fill>
27            <SolidColorBrush Color="#FFFFFF" />
28        </Glyphs.Fill>
29    </Glyphs>
30 </Canvas>

```

31 This markup is rendered as follows:



32

33 *end example]*

34 10.5 <Path> Element

35 The <Path> element specifies a geometry that can be filled with a brush. For more information,
 36 see §11.1.

1 **10.6 <Glyphs> Element**

2 The <Glyphs> element is used to represent a run of uniformly-formatted text from a single
3 font. The <Glyphs> element provides information for accurate rendering and supports search
4 and selection features in XPS Document consumers. For more information, see §12.1.

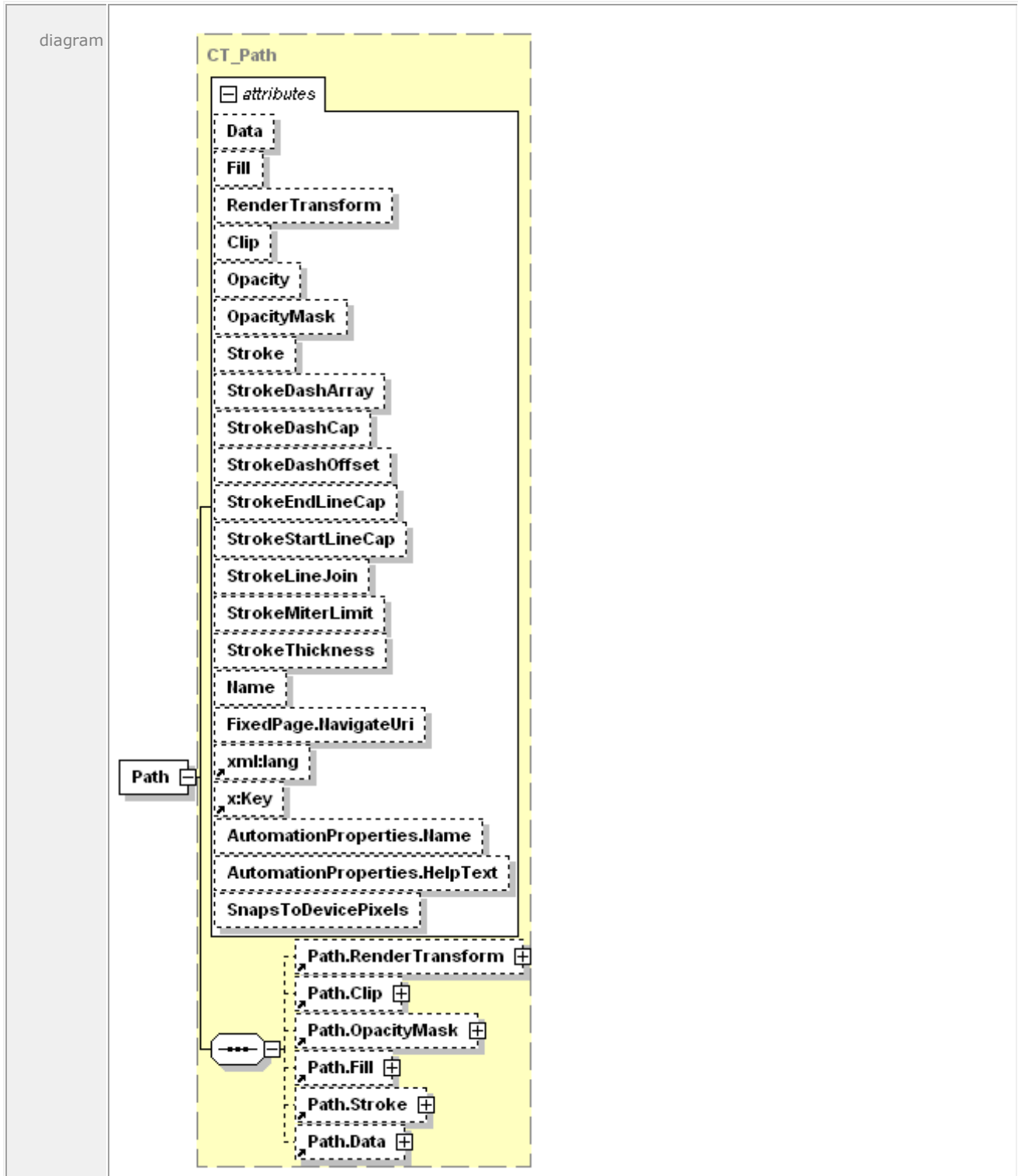
5

1 **11. Graphics**

2 Vector graphics are created using the <Path> element. A full set of properties is available to
3 describe the visual characteristics of the graphic. These characteristics include the fill, opacity,
4 clipping, rendering transformation, and various stroke details including thickness, fill, line join
5 style, line miter limit, line cap style, dash style, and dash cap style. The description of the
6 geometric area of the path (the geometry) is described by the Data property. Raster images are
7 included in fixed page markup by specifying a <Path> element filled with an <ImageBrush>.

1 **11.1 <Path> Element**

2 element **Path**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Data	ST_RscRefAbbrGeomF				Describes the geometry of the path.
	Fill	ST_RscRefColor				Describes the brush used to paint the geometry specified by the Data property of the path.
	RenderTransform	ST_RscRefMatrix				Establishes a new coordinate frame for all attributes of the path and for all child elements of the path, such as the geometry defined by the <Path.Data> property element.
	Clip	ST_RscRefAbbrGeomF				Limits the rendered region of the element.
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the path element. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	OpacityMask	ST_RscRef				Specifies a mask of alpha values that is applied to the path in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
	Stroke	ST_RscRefColor				Specifies the brush used to draw the stroke.
	StrokeDashArray	ST_EvenArrayPos				Specifies the length of dashes and gaps of the outline stroke. These values are specified as multiples of the stroke thickness as a space-separated list with an even number of non-negative values. When a stroke is drawn, the dashes and gaps specified by these values are repeated to cover the length of the stroke. If this attribute is omitted, the stroke is drawn solid, without any

					gaps.
StrokeDashCap	<u>ST_DashCap</u>		Flat		Specifies how the ends of each dash are drawn. Valid values are Flat, Round, Square, and Triangle.
StrokeDashOffset	<u>ST_Double</u>		0.0		Adjusts the start point for repeating the dash array pattern. If this value is omitted, the dash array aligns with the origin of the stroke. Values are specified as multiples of the stroke thickness.
StrokeEndLineCap	<u>ST_LineCap</u>		Flat		Defines the shape of the end of the last dash in a stroke. Valid values are Flat, Square, Round, and Triangle.
StrokeStartLineCap	<u>ST_LineCap</u>		Flat		Defines the shape of the beginning of the first dash in a stroke. Valid values are Flat, Square, Round, and Triangle.
StrokeLineJoin	<u>ST_LineJoin</u>		Miter		Specifies how a stroke is drawn at a corner of a path. Valid values are Miter, Bevel, and Round. If Miter is selected, the value of StrokeMiterLimit is used in drawing the stroke.
StrokeMiterLimit	<u>ST_GEOne</u>		10.0		The ratio between the maximum miter length and half of the stroke thickness. This value is significant only if the StrokeLineJoin attribute specifies Miter.
StrokeThickness	<u>ST_GEZero</u>		1.0		Specifies the thickness of a stroke, in units of the effective coordinate space (includes the path's render transform). The stroke is drawn on top of the boundary of the geometry specified by the <Path> element's Data property. Half of the StrokeThickness extends outside of the geometry specified by the Data property

					and the other half extends inside of the geometry.
Name	<u>ST_Name</u>				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
FixedPage.NavigateUri	xs:anyURI				Associates a hyperlink URI with the element. Can be a relative reference or a URI that addresses a resource that is internal to or external to the package.
xml:lang					Specifies the default language used for the current element and for any child or descendant elements. The language is specified according to RFC 3066.
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.1].
AutomationProperties.Name	xs:string				A brief description of the <Path> for accessibility purposes, particularly if filled with an <ImageBrush>.
AutomationProperties.HelpText	xs:string				A detailed description of the <Path> for accessibility purposes, particularly if filled with an <ImageBrush>.
SnapsToDevicePixels	<u>ST_Boolean</u>				On Anti-aliasing consumers controls if control points snap to the nearest device pixels. Valid values are 'false' and 'true'. Consumers MAY ignore this attribute [O4.1].
annotation	Defines a single graphical effect to be rendered to the page. It paints a geometry with a brush and draws a stroke around it.				

1 The <Path> element is the sole means of adding vector graphics and images to a fixed page. It
 2 defines a single vector graphic to be rendered on a page. Some properties of the <Path>
 3 element are composable, meaning that the markings rendered to the page are determined by a
 4 combination of the property and all of the like-named properties of its parent and ancestor
 5 elements.

6 The Data property contains a geometric description of the area on which to apply a given effect.
 7 This description can take one of two forms: verbose or abbreviated. In the verbose form, the
 8 geometry is described in the <Path.Data> property element using the elements described
 9 in §11.2. In abbreviated form, it is described using abbreviated syntax in the Data attribute. For
 10 more information, see §11.2.3.

11 The <Path.Fill> property element describes the appearance of the area specified by the Data
 12 property. It contains a brush (see §13) that is used to fill the described areas. These can
 13 include a solid color, an image, a gradient, or a vector drawing pattern.

14 The <Path.Stroke> property element describes the appearance of the borders of the shape
 15 specified by the Data property. It also contains a <Brush> element, which is used to fill the
 16 borders according to the stroke properties (such as StrokeThickness). See §18 for detailed
 17 rendering rules of strokes, line caps, and dash caps.

18 If neither Stroke nor Fill properties are specified, the <Path> element has no visible effect.

19 The transparency of the rendered <Path> element is controlled by the Opacity attribute. More
 20 complex transparency descriptions can be defined using the OpacityMask attribute to control the
 21 transparency of the brushes described by the Fill and Stroke properties.

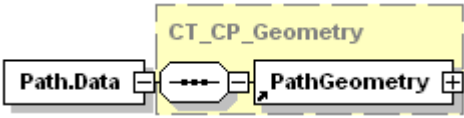
22 Consumers or viewers that perform anti-aliasing MAY “snap” those control points of the path
 23 that are situated on the path bounding box to whole device pixels if the ignorable
 24 SnapsToDevicePixels attribute is specified as true [O4.1].

25 Finally, the path can be cropped by specifying a clipping region in the Clip property, which
 26 describes the geometric area to be preserved. The remainder is not rendered. See §11.2.1 for
 27 how geometries are defined.

28 For details on the Clip, Opacity, OpacityMask, and RenderTransform properties, see §14.

29 **11.1.1 <Path.Data> Element**

30 element **Path.Data**

diagram	 <p>The diagram illustrates the relationship between three elements: Path.Data, CT_CP_Geometry, and PathGeometry. Path.Data is shown as a box on the left, connected by a dashed line to a central box labeled CT_CP_Geometry. This central box is highlighted with a yellow background and is enclosed in a dashed yellow border. CT_CP_Geometry is further connected by a dashed line to a box on the right labeled PathGeometry. All three boxes have small square handles on their right sides, indicating they are part of a sequence or flow.</p>
annotation	Describes the geometry of the path.

31 The <Path.Data> property element describes the geometric area of a path. It contains a single
 32 geometry.

33 *Example 11-1. <Path.Data> usage*

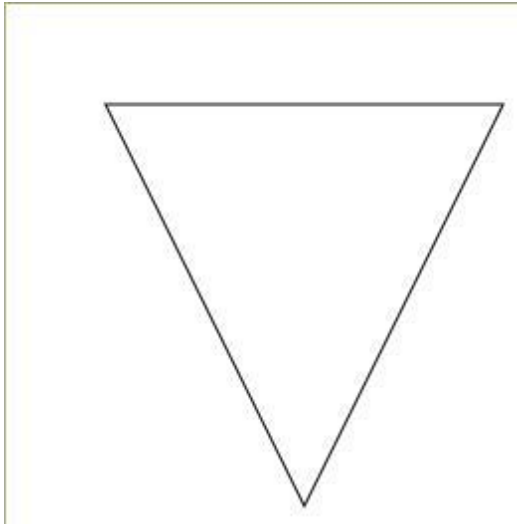
```
34     <Path Stroke="#000000" StrokeThickness="1">
35         <Path.Data>
36             <PathGeometry>
```

```

1      <PathFigure StartPoint="50,50" IsClosed="true">
2          <PolyLineSegment Points="250,50 150,250" />
3      </PathFigure>
4  </PathGeometry>
5  </Path.Data>
6  </Path>

```

7 This markup produces the following results:

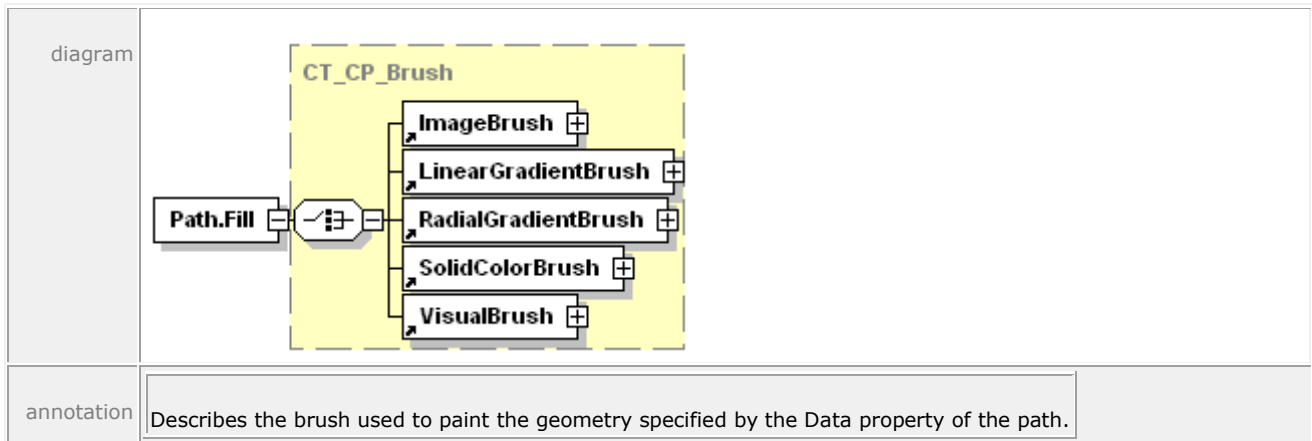


8

9 *end example]*

10 11.1.2 <Path.Fill> Element

11 element **Path.Fill**



12 The <Path.Fill> property element specifies the brush that is used to fill the region described by
 13 the Data property. This can be a solid color, an image, a gradient, or a vector drawing pattern.

1 *Example 11-2. <Path.Fill> usage*

2 In the following markup, the geometry is filled with a solid color:

```

3     <Path>
4         <Path.Fill>
5             <SolidColorBrush Color="#0000FF" />
6         </Path.Fill>
7         <Path.Data>
8             <PathGeometry>
9                 <PathFigure StartPoint="10,10" IsClosed="true">
10                    <PolyLineSegment Points="50,200 100,40 150,200
11                       200,10 100,105" />
12                </PathFigure>
13            </PathGeometry>
14        </Path.Data>
15    </Path>

```

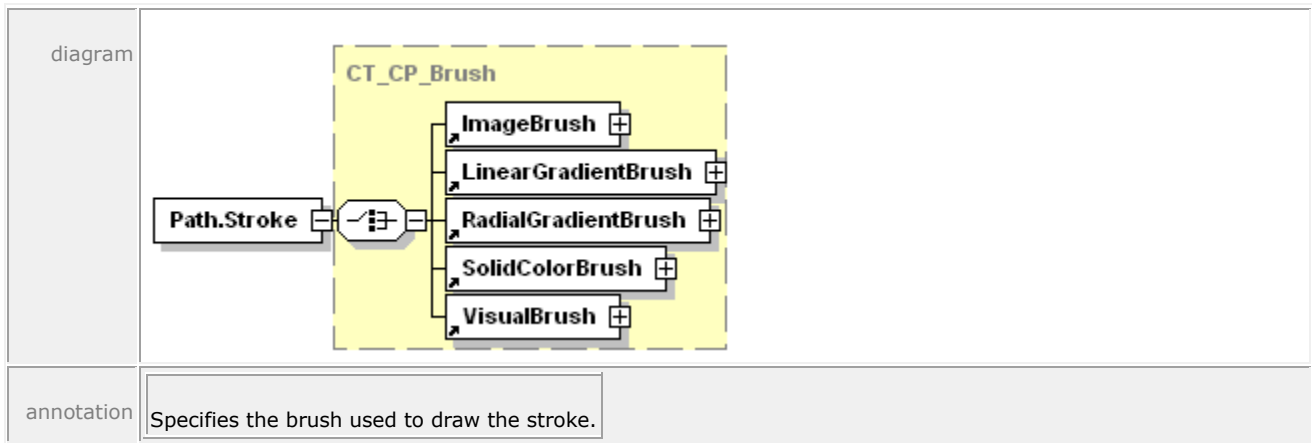
16 This markup produces the following result:



17
18 *end example]*

19 **11.1.3 <Path.Stroke> Element**

20 element **Path.Stroke**



1 The <Path.Stroke> property element describes the border of the path's geometry.
 2 <Path.Stroke> contains a brush. Only those segments of the path figure in the <Path.Data>
 3 element that set the IsStroked attribute to true (the default value if omitted) are stroked. If
 4 IsClosed is set to true, an extra segment will be stroked, connecting the last point in the path
 5 figure with the first point in the path figure.

6 The <Path.Stroke> property element is then used to describe the appearance of the borders of
 7 the shape defined by the Data property. It also contains a brush, which is used to fill the
 8 borders according to the stroke properties (such as StrokeThickness).

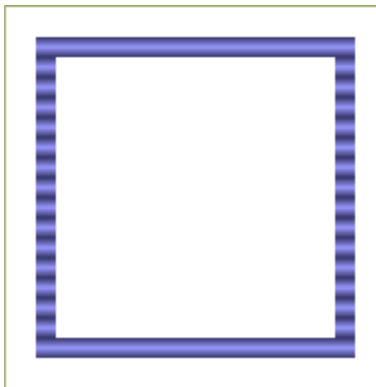
9 For more information, see §18.6.

10 *Example 11-3. <Path.Stroke> usage*

11 The following <Path.Stroke> element uses a gradient brush to fill the border of a box:

```
12     <Path StrokeThickness="10" Data="M 20,20 L 170,20 L 170,170 L 20,170 Z">
13         <Path.Stroke>
14             <LinearGradientBrush MappingMode="Absolute"
15                 StartPoint="0,0" EndPoint="0,5" SpreadMethod="Reflect">
16                 <LinearGradientBrush.GradientStops>
17                     <GradientStop Color="#9999FF" Offset="0.0" />
18                     <GradientStop Color="#333366" Offset="1.0" />
19                 </LinearGradientBrush.GradientStops>
20             </LinearGradientBrush>
21         </Path.Stroke>
22     </Path>
```

23 This markup produces the following results:



24

25 *end example]*

26 11.2 Geometries and Figures

27 Geometries are used to build visual representations of geometric shapes.

28 The smallest atomic unit in a geometry is a segment. Segments can be lines or curves. One or
 29 more segments are combined into a path figure definition. A path figure is a single shape
 30 comprised of continuous segments. One or more path figures collectively define an entire path
 31 geometry. A path geometry MAY define the fill algorithm to be used on the component path
 32 figures [M2.72].

- 1 A single path geometry can be used in the Data property of the <Path> element to describe its
- 2 overall geometry. A path geometry can also be used in the Clip property of the <Canvas>,
- 3 <Path>, or <Glyphs> elements to describe a clipping region.

1 **11.2.1 Geometries**

2 A <PathGeometry> element constitutes a complete geometry definition.

3 **11.2.1.1 <PathGeometry> Element**

4 element **PathGeometry**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Figures	ST_AbbrGeom				Describes the geometry of the path.
	FillRule	ST_FillRule		EvenOdd		Specifies how the intersecting areas of geometric shapes are combined to form a region. Valid values are EvenOdd and NonZero.
	Transform	ST_RscRefMatrix				Specifies the local matrix transformation that is applied to all child and descendant elements of the path geometry before it is used for filling, clipping, or stroking.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.2].
annotation	Contains a set of <PathFigure> elements.					

5 A <PathGeometry> element contains a set of path figures specified either with the Figures
 6 attribute or with a child <PathFigure> element. Producers MUST NOT specify the path figures of
 7 a geometry with both the Figures attribute and a child <PathFigure> element [M4.3].

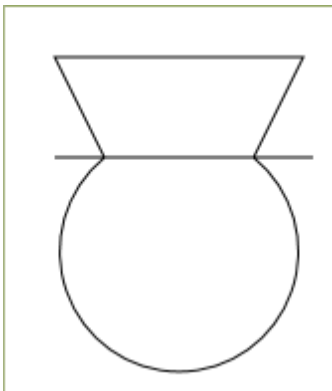
8 The union of the path figures defines the interior of the path geometry according to the FillRule
 9 attribute as described in §11.2.1.2.

```

1 Example 11-4. <PathGeometry> usage
2   <Path Stroke="#000000">
3     <Path.Data>
4       <PathGeometry>
5         <PathFigure StartPoint="25,75">
6           <PolyLineSegment Points="150,75 50,75" />
7         </PathFigure>
8         <PathFigure StartPoint="50,75" IsClosed="true">
9           <ArcSegment
10            Size="60,60"
11            RotationAngle="0"
12            IsLargeArc="true"
13            SweepDirection="Counterclockwise"
14            Point="125,75" />
15         </PathFigure>
16         <PathFigure StartPoint="50,75" IsClosed="true">
17           <PolyLineSegment Points="25,25 150,25 125,75" />
18         </PathFigure>
19       </PathGeometry>
20     </Path.Data>
21   </Path>

```

22 This markup produces the following results:



23

24 *end example]*

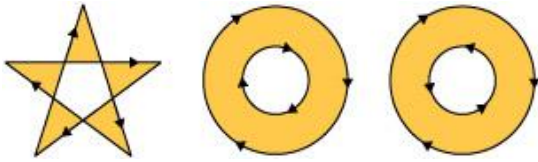
25 11.2.1.2 FillRule Attribute

26 The FillRule attribute specifies a fill algorithm. The fillable area of a geometry is defined by
 27 taking all of the contained path figures and applying the fill algorithm to determine the enclosed
 28 area. Fill algorithms determine how the intersecting areas of geometric shapes are combined to
 29 form a region.

30 11.2.1.2.1 EvenOdd Fill Algorithm

31 This rule determines the "insideness" of a point on the canvas by drawing a ray from the point
 32 to infinity in any direction and counting the number of segments from the given shape that the
 33 ray crosses. If this number is odd, the point is inside; if it is even, the point is outside. This is
 34 the default rule used throughout XPS Document markup.

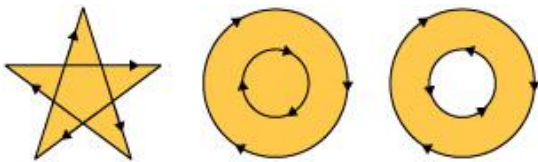
1 *Figure 11-1. Fill using EvenOdd algorithm*



3 **11.2.1.2.2 NonZero Fill Algorithm**

4 This rule determines the “insideness” of a point on the canvas by drawing a ray from the point
 5 to infinity in any direction and then examining the places where a segment of the shape crosses
 6 the ray. Starting with a count of zero, add one each time a segment crosses the ray from left to
 7 right and subtract one each time a path segment crosses the ray from right to left. After
 8 counting the crossings, if the result is zero then the point is outside the path; otherwise, it is
 9 inside.

10 *Figure 11-2. Fill using NonZero algorithm*



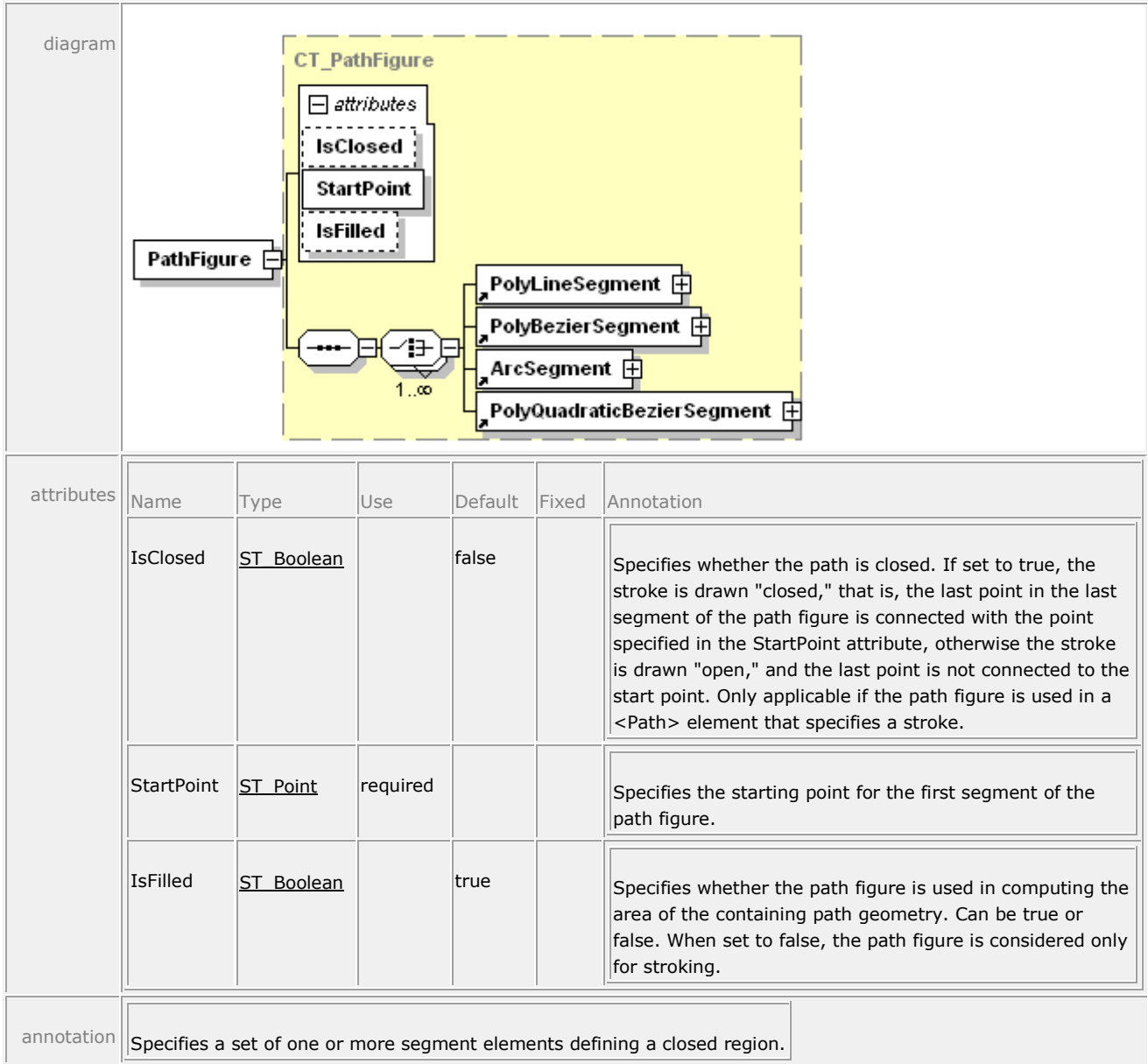
12 **11.2.1.3 Figures Attribute**

13 The <PathGeometry> element’s Figures attribute can be used to describe the path figures the
 14 geometry contains using abbreviated syntax (see §11.2.3) with the exception that the FillRule
 15 command MUST NOT be used [M2.72].

1 **11.2.2 Figures**

2 **11.2.2.1 <PathFigure> Element**

3 element **PathFigure**



4 A <PathFigure> element is composed of a set of one or more line or curve segments. The
 5 segment elements define the shape of the path figure. The initial point of the first segment
 6 element is specified as the StartPoint attribute of the path figure. The last point of each segment
 7 element is the first point of the following segment element.

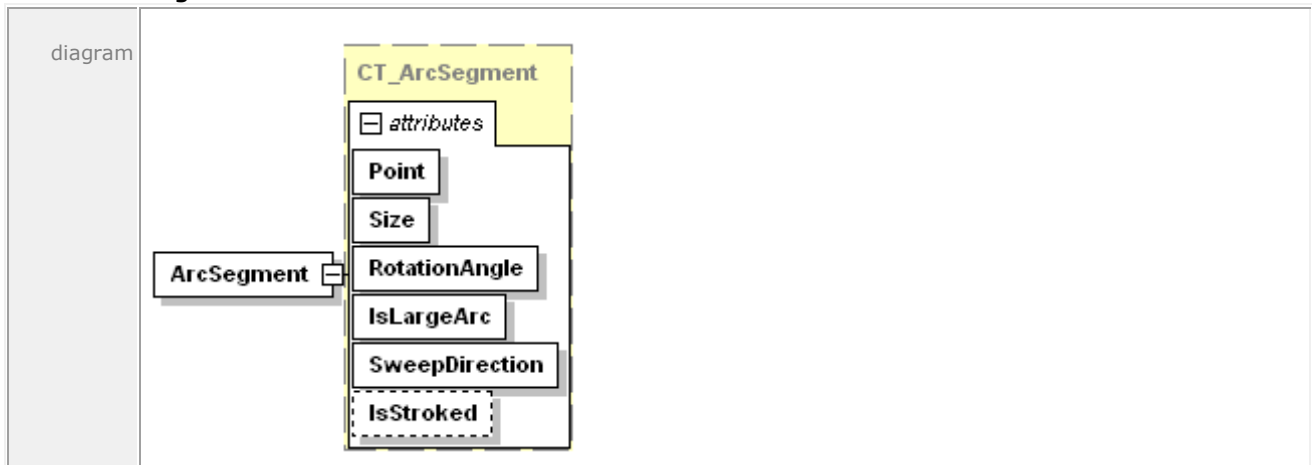
1 Segment elements are:

- 2 • <ArcSegment>
- 3 • <PolyBezierSegment>
- 4 • <PolyLineSegment>
- 5 • <PolyQuadraticBezierSegment>

6 Line segments and curve segments SHOULD NOT be specified as zero-length [S4.1]. If they are
 7 specified as zero-length, [they are not drawn. For full details of the behavior in cases such as](#)
 8 [those involving line caps, see §18.](#) ~~nothing is drawn, even if line caps would normally be visible.~~
 9 ~~For further details and exceptions, see §18.~~

10 **11.2.2.2 <ArcSegment> Element**

11 element **ArcSegment**



attributes	Name	Type	Use	Default	Fixed	Annotation
		Point	<u>ST_Point</u>	required		
	Size	<u>ST_PointGE0</u>	required			Specifies the x and y radius of the elliptical arc as an x,y pair.
	RotationAngle	<u>ST_Double</u>	required			Indicates how the ellipse is rotated relative to the current coordinate system.
	IsLargeArc	<u>ST_Boolean</u>	required			Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.
	SweepDirection	<u>ST_SweepDirection</u>	required			Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.
	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment

						of the path is drawn. Can be true or false.
annotation	Represents an elliptical arc between two points.					

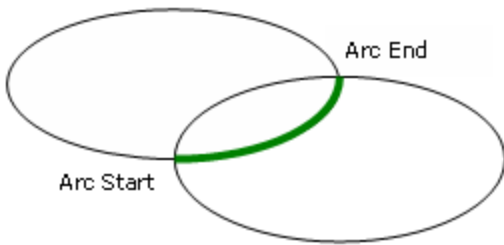
1 The <ArcSegment> element describes an elliptical arc. It is geometrically defined by the
 2 intersection of two ellipses that have the same x radius and y radius. The ellipses intersect at
 3 the starting and ending points of the arc.

4 *Table 11-1. Arc segment definition*

Term	Description
Starting Point	Implicitly defined by the previous point in the path figure definition.
Ending Point	Specified by the Point attribute.
Arc Size	Defined by the Size attribute. This value consists of the comma-delimited x and y radii of the ellipses that will be used to define the arc. [Example: "100,50" end example]
Rotation Angle	Specified by the RotationAngle attribute, this determines how the ellipses defining the arc are rotated with respect to the x axis, in degrees. Positive values are clockwise and negative values are counter-clockwise.
Large Arc Flag	Specified by the IsLargeArc attribute, this flag indicates which of the arc pairs created by the intersecting ellipses to use. When the flag is true, it uses the larger arc (arc length >= 180°), and when it is false it uses the smaller arcs (arc length < 180°).
Sweep Direction	Specified by the SweepDirection attribute, this flag determines which of the two possible arcs (selected by the Large Arc Flag) is used. Beginning at the starting point, one arc proceeds in the positive (clockwise) direction, while the other proceeds in the negative (counter-clockwise) direction.

1 *Figure 11-3. Arc choice A*

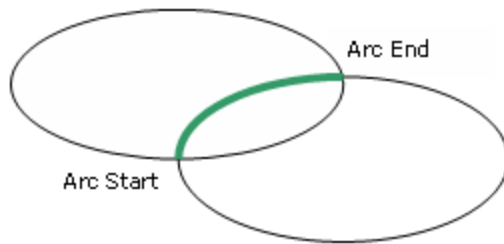
2 `IsLargeArc = false; SweepDirection = Counterclockwise`



3

4 *Figure 11-4. Arc choice B*

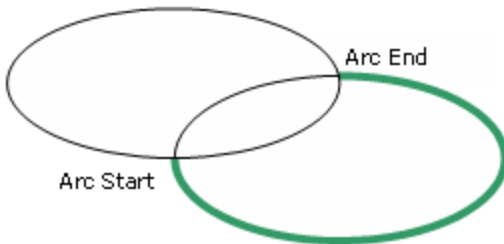
5 `IsLargeArc = false; SweepDirection = Clockwise`



6

7 *Figure 11-5. Arc choice C*

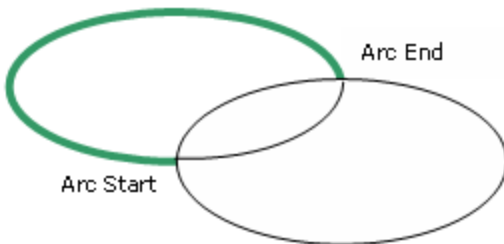
8 `IsLargeArc = true; SweepDirection = Counterclockwise`



9

10 *Figure 11-6. Arc choice D*

11 `IsLargeArc = true; SweepDirection = Clockwise`



12

13 *Example 11-5. <ArcSegment> usage*

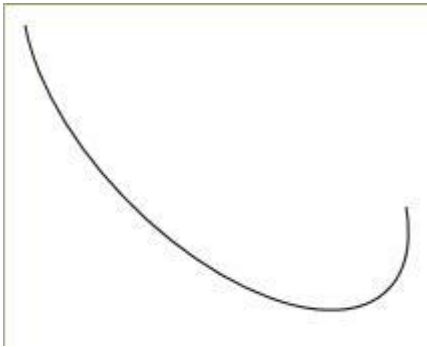
```
14 <Path Stroke="#000000" StrokeThickness="1">
15 <Path.Data>
```

```

1      <PathGeometry>
2          <PathFigure StartPoint="10,10">
3              <ArcSegment
4                  Size="100,50"
5                  RotationAngle="45"
6                  IsLargeArc="true"
7                  SweepDirection="Counterclockwise"
8                  Point="200,100" />
9          </PathFigure>
10     </PathGeometry>
11 </Path.Data>
12 </Path>

```

13 This markup generates the following arc:



14

15 *end example]*

16 **11.2.2.2.1 Out-of-Range Attributes**

17 The following guidelines are followed when encountering incompatible attribute values on an
18 <ArcSegment> element:

- 19 • If the arc is impossible to render given the combination of radii specified in the Size
20 attribute and the angle of rotation specified in the RotationAngle attribute, the ellipses are
21 scaled equally until there is exactly one solution that satisfies the arc requirements to
22 pass through the specified Point attribute.
- 23 • If the Point attribute is the same as the previous point in the path figure, the segment is
24 omitted.
- 25 • If either the x or y radius in the Size attribute is 0, the segment is rendered as a poly line
26 segment with a single line segment to the x,y coordinates specified by the Point
27 attribute.
- 28 • The x or y radius in the Size attribute MUST NOT be negative [M2.72].
- 29 • If the RotationAngle value is greater than 360, it is replaced by the value of the
30 RotationAngle modulo 360. If it is less than 0, it is replaced with a value normalized to the
31 range 0–360.

1 **11.2.2.3 <PolyBezierSegment> Element**2 element **PolyBezierSegment**

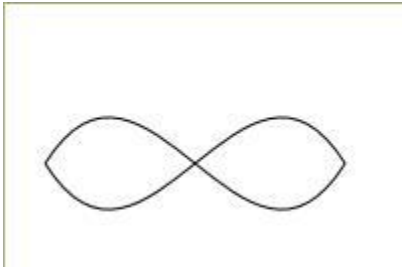
diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<u>ST_Points</u>	required			Specifies control points for multiple Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.
	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.
annotation	A series of Bézier segments.					

3 The <PolyBezierSegment> element describes a set of cubic Bézier curves. Bézier curves are
4 drawn from the previous point in the path figure or the previous Bézier curve in the segment
5 and terminate at the third point (x_{3n}, y_{3n}) in the Points attribute (where n is the curve being
6 drawn). The tangents and curvature of each Bézier curve are controlled by the first two control
7 points (x_{3n-2}, y_{3n-2} and x_{3n-1}, y_{3n-1}) in the Points attribute. The Points attribute contains a multiple
8 of three whitespace-delimited pairs of comma-delimited x,y values.

9 *Example 11-6. <PolyBezierSegment> usage*

```
10 <Path Stroke="#000000" StrokeThickness="1">
11 <Path.Data>
12 <PathGeometry>
13 <PathFigure StartPoint="20,80">
14 <PolyBezierSegment Points="70,0 120,160 170,80 120,0 70,160
15 20,80" />
16 </PathFigure>
17 </PathGeometry>
18 </Path.Data>
19 </Path>
```

1 This markup generates the following results:



2
3 *end example]*

4 **11.2.2.4 <PolyLineSegment> Element**

5 element **PolyLineSegment**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<u>ST_Points</u>	required			Specifies a set of coordinates for the multiple segments that define the poly line segment. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.
	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.
annotation	Specifies a set of points between which lines are drawn.					

6 The <PolyLineSegment> element describes a polygonal drawing containing an arbitrary number
7 of individual vertices. The Points attribute defines the vertices and contains whitespace-
8 delimited pairs of comma-delimited x,y values.

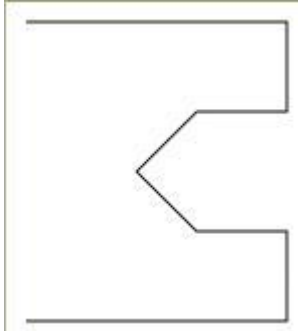
9 *Example 11-7. <PolyLineSegment> usage*

```

10 <Path Stroke="#000000" StrokeThickness="1">
11 <Path.Data>
12 <PathGeometry>
13 <PathFigure StartPoint="10,10">
14 <PolyLineSegment Points="140,10 140,55 95,55 65,85 95,115
15 140,115 140,160 10,160" />
16 </PathFigure>
17 </PathGeometry>
18 </Path.Data>
    
```


1 </Path>

2 This markup produces the following figure:



3

4 *end example]*

5 **11.2.2.5 <PolyQuadraticBezierSegment> Element**

6 element **PolyQuadraticBezierSegment**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<u>ST_Points</u>	required			Specifies control points for multiple quadratic Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.
	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.
annotation	A series of quadratic Bézier segments.					

7 The <PolyQuadraticBezierSegment> element describes a set of quadratic Bézier curves from
 8 the previous point in the path figure through a set of vertices, using specified control points.

9 The Points attribute defines an off-curve control point (x_{2n-1},y_{2n-1}) followed by the end point
 10 (x_{2n},y_{2n}) for each quadratic Bézier curve (where n represents the quadratic Bézier curve). The
 11 Points attribute contains a multiple of two whitespace-delimited pairs of comma-delimited x,y
 12 values.

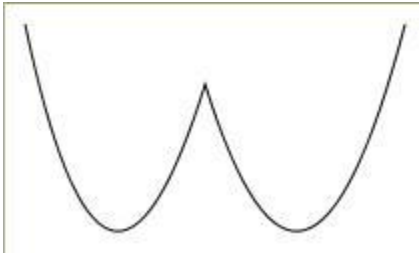
1 *Example 11-8. <PolyQuadraticBezierSegment> usage*

```

2     <Path Stroke="#000000" StrokeThickness="1">
3         <Path.Data>
4             <PathGeometry>
5                 <PathFigure StartPoint="10,10">
6                     <PolyQuadraticBezierSegment Points="50,200 100,40 150,200
7                         200,10" />
8                 </PathFigure>
9             </PathGeometry>
10        </Path.Data>
11    </Path>

```

12 This markup produces the following curve:



13

14 *end example]*

15 **11.2.2.6 Closed <PathFigure>**

16 If the `IsClosed` attribute of the `<PathFigure>` element is set to `true`, a straight line is drawn from
 17 the last point in the last segment of the `<PathFigure>` element to the `StartPoint` attribute of the
 18 `<PathFigure>` element. If the `IsClosed` attribute is omitted, its default setting is `false`.

19 `<PathFigure>` elements used in filled `<Path>` elements or as `Clip` attributes are implicitly
 20 closed.

21 *Example 11-9. Closed <PathFigure> usage*

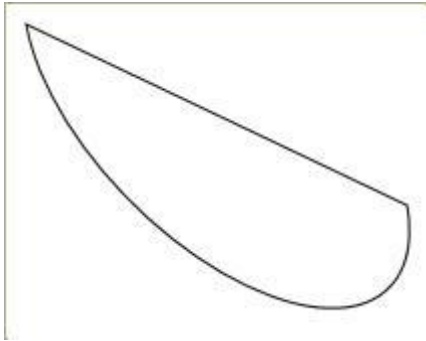
22 The following markup shows the arc segment as shown in Example 11-5 with the `IsClosed`
 23 attribute of the `<PathFigure>` element set to `true`.

```

24     <Path Stroke="#000000" StrokeThickness="1">
25         <Path.Data>
26             <PathGeometry>
27                 <PathFigure StartPoint="10,10" IsClosed="true">
28                     <ArcSegment
29                         Size="100,50"
30                         RotationAngle="45"
31                         IsLargeArc="true"
32                         SweepDirection="Counterclockwise"
33                         Point="200,100" />
34                 </PathFigure>
35             </PathGeometry>
36        </Path.Data>
37    </Path>

```

1 This markup generates the following figure:



2
3 *end example]*

4 **11.2.3 Abbreviated Geometry Syntax**

5 Abbreviated geometry syntax MAY be used to specify a geometry of one or more figures
6 comprised of multiple segments [M2.72]. A geometry is specified with an optional FillRule
7 command (not allowed in the Figures attribute of a <PathGeometry> element) followed by one
8 or more figure definitions. Figure definitions are specified with a Move command, a set of
9 drawing commands to create segments, and an optional Close command to create a closing
10 segment. Drawing commands include:

- 11 • Line
- 12 • Horizontal Line
- 13 • Vertical Line
- 14 • Cubic Bézier Curve
- 15 • Quadratic Bézier Curve
- 16 • Smooth Cubic Bézier Curve
- 17 • Elliptical Arc

18 A command is represented by a single letter and is followed by zero or more whitespace
19 characters, which are followed by command parameters. Parameters are whitespace-delimited.
20 Points are specified as a comma-delimited pair with zero or more whitespace characters.

21 Uppercase letters denote absolute values and lowercase letters denote relative values. When
22 relative coordinate values are specified, each coordinate pair expresses an offset relative to the
23 current endpoint (the previous command's terminating coordinate pair). If a relative value is
24 used for the first Move command, the current endpoint is, by definition, 0,0.

25 If a relative value is used following a Close command, the current endpoint is the first point of
26 the previous figure.

27 If entering more than one drawing command of the same type sequentially, the duplicate
28 command entry MAY be omitted [M2.72]. [*Example*: "L 100,200 300,400" is equivalent to "L
29 100,200 L 300,400". *end example*] The current endpoint is determined as though each
30 command appeared individually.

31 Values specifying coordinates can be real numbers.

32 For more information, see §F.

1 Table 11-2. Commands

Name	Syntax	Description	Non-Abbreviated Equivalent
FillRule	F fFillRule	Establishes the fill rule that should be used for this geometry. A value of 0 is equivalent to a FillRule value of EvenOdd; a value of 1 is equivalent to a FillRule value of NonZero. The default value if this command is omitted is 0. This command MUST appear only as the first command in the abbreviated geometry syntax [M2.72]. This command MUST NOT be specified in the value of the Figures attribute of the <PathGeometry> element [M2.72]. [Example: F 0 end example]	<PathGeometry> FillRule attribute
Move	M x, y or m x, y	Establishes a new current endpoint. Every geometry MAY specify one or more figures, and MAY be preceded by a FillRule command where allowed [M2.72]. The first figure in a geometry MUST begin with a Move command [M2.72]. Subsequent Move commands indicate the start of a new figure but MAY be omitted, indicating the current endpoint for the subsequent figure is the same as the end point of the previous figure [M2.72]. [Example: M 1.0,1.5 end example]	<PathFigure> StartPoint attribute
Line	L x, y or l x, y	Draws a straight line from the current point to the specified point. [Example: L 20,30 end example]	<PolyLineSegment> element
Horizontal Line	H x or h x	Draws a horizontal line from the current endpoint to the specified x coordinate. [Example: H 90 end example]	<PolyLineSegment> element
Vertical Line	V y or v y	Draws a vertical line from the current endpoint to the specified y coordinate.	<PolyLineSegment> element

Name	Syntax	Description	Non-Abbreviated Equivalent
		[<i>Example: v 90 end example</i>]	
Cubic Bézier Curve	C x_1, y_1 x_2, y_2 x_3, y_3 or c x_1, y_1 x_2, y_2 x_3, y_3	Draws a cubic Bézier curve from the current endpoint to the specified point (x_3, y_3) using the two specified control points (x_1, y_1 and x_2, y_2). The first control point determines the initial direction (tangent) of the curve, and the second determines the terminating direction (tangent) of the curve. [<i>Example: c 100,200 200,400 300,200 end example</i>]	<PolyBezierSegment> element
Quadratic Bézier Curve	Q x_1, y_1 x_2, y_2 or q x_1, y_1 x_2, y_2	Draws a quadratic Bézier curve from the current endpoint to the specified point (x_2, y_2) using the specified control point (x_1, y_1). [<i>Example: q 100,200 300,200 end example</i>]	<PolyQuadraticBezierSegment> element
Smooth Cubic Bézier Curve	S x_1, y_1 x_2, y_2 or s x_1, y_1 x_2, y_2	Draws a cubic Bézier curve from the current endpoint to the specified point (x_2, y_2). The first control point is assumed to be the reflection of the second control point of the previous command, relative to the current endpoint. If there is no previous command or if the previous command was not a Cubic Bézier Curve command or Smooth Cubic Bézier Curve command, the first control point is assumed to be coincident with the current endpoint. The second control point is specified by x_1, y_1 . [<i>Example: s 100,200 200,300 end example</i>]	<PolyBezierSegment> element
Elliptical Arc	A x_r, y_r r_x fArc fSweep x, y or a x_r, y_r r_x fArc fSweep x, y	Draws an elliptical arc from the current endpoint to the specified point (x, y). The size and orientation of the ellipse are defined by x_r, y_r . r_x, x_r defines the x radius, y_r defines the y radius, and r_x defines the x-axis rotation in degrees, which indicates how the	<ArcSegment> element

Name	Syntax	Description	Non-Abbreviated Equivalent
		<p>ellipse is rotated relative to the current coordinate system. The center of the ellipse is calculated automatically.</p> <p>In most situations, four different arcs satisfy the specified constraints. f_{Arc} and f_{Sweep} indicate which arc to use.</p> <p>Of the four candidate arc sweeps, two represent large arcs with sweeps of 180° or greater, and two represent smaller arcs with sweeps less than 180°.</p> <p>If f_{Arc} is 1, one of the two larger arc sweeps is chosen. If f_{Arc} is 0, one of the smaller arc sweeps is chosen. No other values of f_{Arc} are valid.</p> <p>If f_{Sweep} is 1, the arc is drawn in a positive-angle (clockwise) direction. If f_{Sweep} is 0, the arc is drawn in a negative-angle (counter-clockwise) direction. No other values of f_{Sweep} are valid. [Example: a 200,70 10 0 1 100,100 end example]</p>	
Close	z or z	<p>Draws a straight line from the current endpoint to the first point of the current figure and then ends the figure.</p> <p>If the command following a Close command is a Move command, the Move command specifies the initial point of the next figure. Otherwise, the next figure starts at the same initial point as the current figure.</p>	<PathFigure> IsClosed attribute

1 *Example 11-10. A path described using abbreviated syntax*

2 The following markup demonstrates a simple path, which is drawn using the abbreviated
3 syntax:

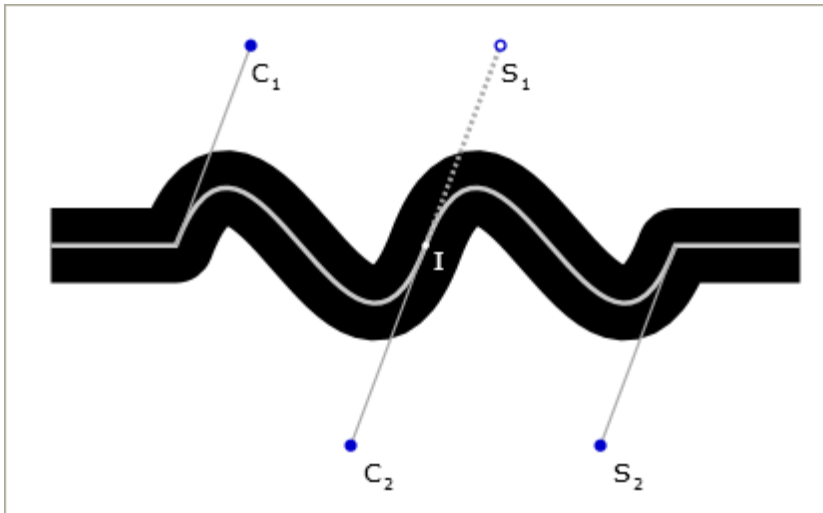
1 <Path Stroke="#000000" Data="M 100,100 L 300,100 L 200,300 z" />
 2 *end example]*

3 **11.2.3.1 Smooth Bézier Curve Abbreviated Syntax**

4 Smooth Bézier curves specified with the abbreviated geometry syntax are basic cubic Bézier
 5 curves with an implied first control point. This control point is coincident with the endpoint of
 6 the previous segment unless the previous segment is also a Bézier curve. In this case, the first
 7 control point of the smooth Bézier curve is a reflection of the second control point of the
 8 previous curve segment around the start point of the smooth Bézier curve segment, as shown
 9 below.

10 *Example 11-11. Smooth Bézier curve*

11 In the following example, C_1 and C_2 represent the first and second control points of the first
 12 cubic Bézier curve segment, respectively. S_1 represents the implied first control point of the
 13 smooth Bézier curve segment. S_2 represents the specified control point of the smooth Bézier
 14 curve segment. I represents the inflection point around which control point S_1 is derived from
 15 control point C_2 .



16

17 The above diagram is generated with the following markup:

```

18     <Canvas RenderTransform="1.25,0,0,1.25,-40,20" >
19       <!-- Main Path -->
20       <Path Stroke="#000000" StrokeThickness="30" StrokeLineJoin="Round"
21       Data="M50,80 L100,80 C130,0 170,160 200,80 S270,160 300,80 L350,80"/>
22       <Path Stroke="#CCCCCC" StrokeThickness="2"
23       Data="M50,80 L100,80 C130,0 170,160 200,80 S270,160 300,80 L350,80"/>
24       <!-- C1 -->
25       <Path Stroke="#AAAAAA" StrokeThickness="1" Data="M 100,80 L 130,0" />
26       <Path Stroke="#0000CC" StrokeThickness="5" StrokeStartLineCap="Round"
27       StrokeEndLineCap="Round" Data="M 130,0 L 130,0" />
28       <Glyphs Fill="#000000" UnicodeString="C" OriginX="130" OriginY="15"
29       FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="10"
30       />
31       <Glyphs Fill="#000000" UnicodeString="1" OriginX="138" OriginY="18"
32       FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="6"
33       />

```

```

1      <!-- C2 -->
2      <Path Stroke="#AAAAAA" Data="M 200,80 L 170,160" />
3      <Path Stroke="#0000CC" StrokeThickness="5" StrokeStartLineCap="Round"
4          StrokeEndLineCap="Round" Data="M 170,160 L 170,160" />
5      <Glyphs Fill="#000000" UnicodeString="C" OriginX="175"
6          OriginY="175" FontUri="../Resources/Fonts/Verdana.ttf"
7          FontRenderingEmSize="10" />
8      <Glyphs Fill="#000000" UnicodeString="2" OriginX="183"
9          OriginY="178" FontUri="../Resources/Fonts/Verdana.ttf"
10         FontRenderingEmSize="6" />
11     <!-- S1 -->
12     <Path Stroke="#AAAAAA" StrokeThickness="2"
13         StrokeDashArray="0.75 0.75" Data="M 200,80 L 230,0" />
14     <Path Stroke="#0000CC" StrokeThickness="5" StrokeStartLineCap="Round"
15         StrokeEndLineCap="Round" Data="M 230,0 L 230,0" />
16     <Path Stroke="#FFFFFF" StrokeThickness="3" StrokeStartLineCap="Round"
17         StrokeEndLineCap="Round" Data="M 230,0 L 230,0" />
18     <Glyphs Fill="#000000" UnicodeString="S" OriginX="230" OriginY="15"
19         FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="10"
20     />
21     <Glyphs Fill="#000000" UnicodeString="1" OriginX="238" OriginY="18"
22         FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="6"
23     />
24     <!-- S2 -->
25     <Path Stroke="#AAAAAA" StrokeThickness="1" Data="M 300,80 L 270,160"
26     />
27     <Path Stroke="#0000CC" StrokeThickness="5" StrokeStartLineCap="Round"
28         StrokeEndLineCap="Round" Data="M 270,160 L 270,160" />
29     <Glyphs Fill="#000000" UnicodeString="S" OriginX="275" OriginY="175"
30         FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="10"
31     />
32     <Glyphs Fill="#000000" UnicodeString="2" OriginX="283" OriginY="178"
33         FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="6"
34     />
35     <!-- Inflection -->
36     <Path Stroke="#FFFFFF" StrokeThickness="3" StrokeStartLineCap="Round"
37         StrokeEndLineCap="Round" Data="M 200,80 L 200,80" />
38     <Glyphs Fill="#FFFFFF" UnicodeString="I" OriginX="203" OriginY="90"
39         FontUri="../Resources/Fonts/Verdana.ttf" FontRenderingEmSize="10"
40     />
41 </Canvas>

```

42 *end example]*

43 11.2.3.2 Relative Commands and Curve Control Points

44 When using relative (lowercase) commands with the abbreviated geometry syntax, each control
45 point and end point are individually specified relative to the start point of that segment.

46 *Example 11–12. Relative commands and curves*

47 The following markup describes a simple shape using cubic Bézier curves:

```

48     <Path Stroke="#000000" Data="M 50,20 L 150,20 C 250,75 170,130 120,100
49         C 70,70 90,110 130,160 Q 0,150 50,20" />

```

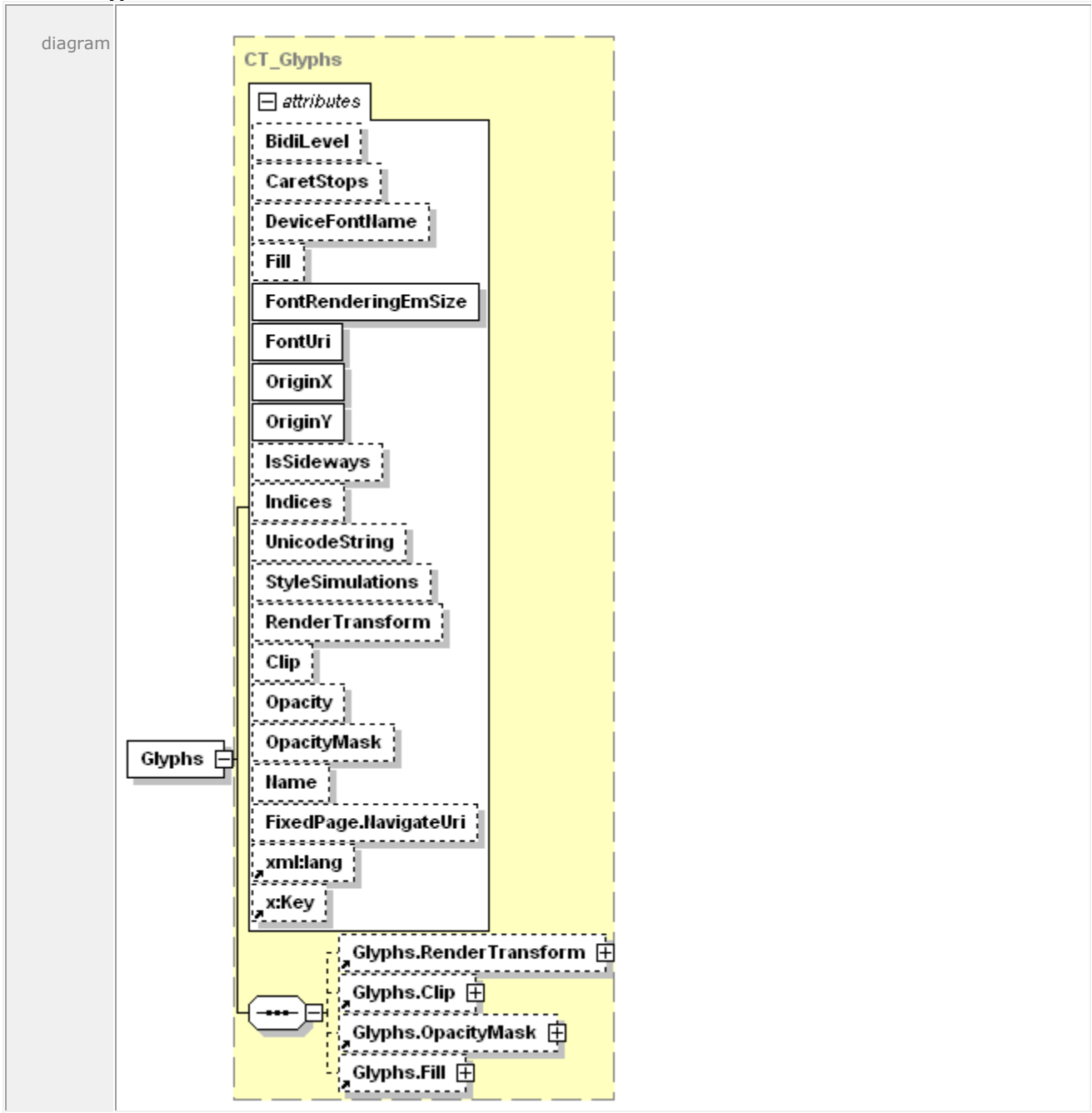

- 1 This markup describes the same shape, using relative commands:
- 2 <Path Stroke="#000000" Data="M 50,20 l 100,0 c 100,55 20,110 -30,80
- 3 c -50,-30 -30,10 10,60 q -130,-10 -80,-140" />
- 4 *end example]*

1 **12. Text**

2 A run of text sharing the same characteristics is represented by a <Glyphs> element. Text runs
3 are broken by line advances and formatting changes. The set of properties on the <Glyphs>
4 element allows for a complete description of the glyph characteristics, such as the fill and
5 opacity, as well as clipping information. The <Glyphs> element allows specification of a Unicode
6 string and supports bidirectional and vertical text.

1 **12.1 <Glyphs> Element**

2 element **Glyphs**



attributes	Name	Type	Use	Default	Fixed	Annotation
	BidiLevel			0		Specifies the Unicode algorithm bidirectional nesting level. Even values imply left-to-right layout,

					<p>odd values imply right-to-left layout. Right-to-left layout places the run origin at the right side of the first glyph, with positive advance widths (representing advances to the left) placing subsequent glyphs to the left of the previous glyph. Valid values range from 0 to 61, inclusive.</p>
CaretStops	<u>ST_CaretStops</u>				<p>Identifies the positions within the sequence of Unicode characters at which a text-selection tool can place a text-editing caret. Potential caret-stop positions are identified by their indices into the UTF-16 code units represented by the UnicodeString attribute value. When this attribute is missing, the text in the UnicodeString attribute value MUST be interpreted as having a caret stop between every Unicode UTF-16 code unit and at the beginning and end of the text [M5.1].</p> <p>The value SHOULD indicate that the caret cannot stop in front of most combining marks or in front of the second UTF-16 code unit of UTF-16 surrogate pairs [S5.1].</p>
DeviceFontName	<u>ST_UnicodeString</u>				<p>Uniquely identifies a specific device font. The identifier is typically defined by a hardware vendor or font vendor.</p>
Fill	<u>ST_RscRefColor</u>				<p>Describes the brush used to fill the shape of the rendered glyphs.</p>
FontRenderingEmSize	<u>ST_GEZero</u>	required			<p>Specifies the font size in drawing surface units, expressed as a float in units of the effective coordinate space. A value of 0 results in no visible text.</p>
FontUri	xs:anyURI	required			<p>The URI of the physical font from which all glyphs in the run are drawn. The URI MUST reference a font contained in the package [M2.1]. If the physical font referenced is a TrueType Collection (containing multiple font faces),</p>

					the fragment portion of the URI is a 0-based index indicating which font face of the TrueType Collection should be used.
OriginX	<u>ST_Double</u>	required			Specifies the x coordinate of the first glyph in the run, in units of the effective coordinate space. The glyph is placed so that the leading edge of its advance vector and its baseline intersect with the point defined by the OriginX and OriginY attributes.
OriginY	<u>ST_Double</u>	required			Specifies the y coordinate of the first glyph in the run, in units of the effective coordinate space. The glyph is placed so that the leading edge of its advance vector and its baseline intersect with the point defined by the OriginX and OriginY attributes.
IsSideways	<u>ST_Boolean</u>		false		Indicates that a glyph is turned on its side, with the origin being defined as the top center of the unturned glyph.
Indices	<u>ST_Indices</u>				Specifies a series of glyph indices and their attributes used for rendering the glyph run. If the UnicodeString attribute specifies an empty string (" " or "{}") and the Indices attribute is not specified or is also empty, a consumer MUST generate an error [M5.2].
UnicodeString	<u>ST_UnicodeString</u>				Contains the string of text rendered by the <Glyphs> element. The text is specified as Unicode code points.
StyleSimulations	<u>ST_StyleSimulations</u>		None		Specifies a style simulation. Valid values are None, ItalicSimulation, BoldSimulation, and BoldItalicSimulation.
RenderTransform	<u>ST_RscRefMatrix</u>				Establishes a new coordinate frame for the glyph run specified by the <Glyphs> element. The render transform affects clip, opacity

					mask, fill, x origin, y origin, the actual shape of individual glyphs, and the advance widths. The render transform also affects the font size and values specified in the Indices attribute.
Clip	<u>ST_RscRefAbbrGeomF</u>				Limits the rendered region of the element. Only portions of the <Glyphs> element that fall within the clip region (even partially clipped characters) produce marks on the page.
Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the glyph element. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
OpacityMask	<u>ST_RscRef</u>				Specifies a mask of alpha values that is applied to the glyphs in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
Name	<u>ST_Name</u>				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
FixedPage.NavigateUri	xs:anyURI				Associates a hyperlink URI with the element. May be a relative reference or a URI that addresses a resource that is internal to or external to the package.
xml:lang					Specifies the default language used for the current element. The language is specified according to RFC 3066.
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M5.3].

annotation	Represents a run of text from a single font.
------------	--

- 1 The <Glyphs> element represents a run of uniformly-formatted text from a single font. It
- 2 provides information necessary for accurate rendering and supports search and selection
- 3 features in viewing consumers.

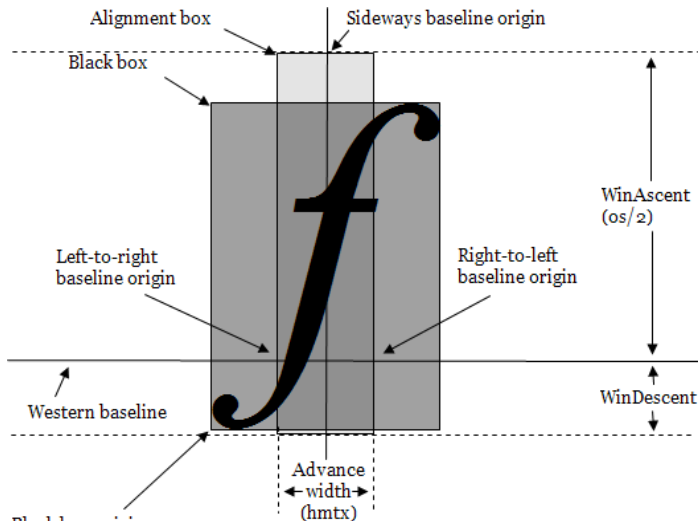
- 4 If the Fill property is not specified, the <Glyphs> element has no visible effect.

- 5 Some properties of the <Glyphs> element are composable, meaning that the markings
- 6 rendered to the page are determined by a combination of the property and all the like-named
- 7 properties of the <Glyphs> element's parent and ancestor elements. For details, see §14.

1 **12.1.1 Glyph Metrics**

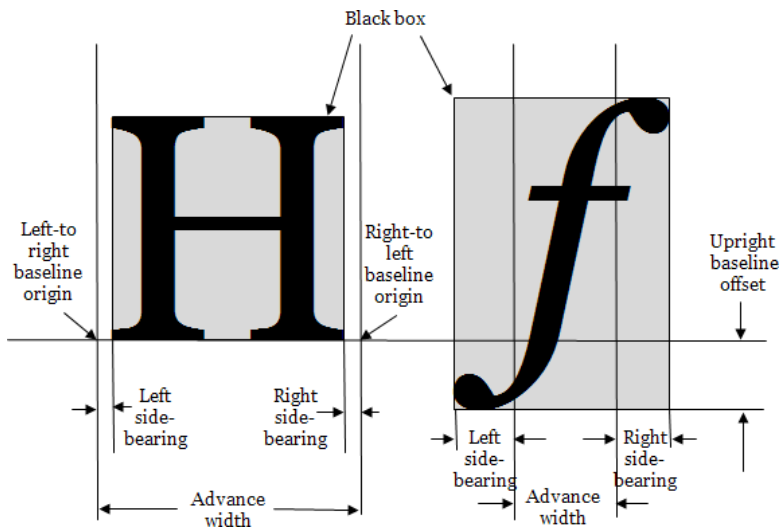
2 Each glyph defines metrics that specify how it aligns with other glyphs. The metrics are
 3 illustrated below.

4 *Figure 12-1. Glyph metrics*

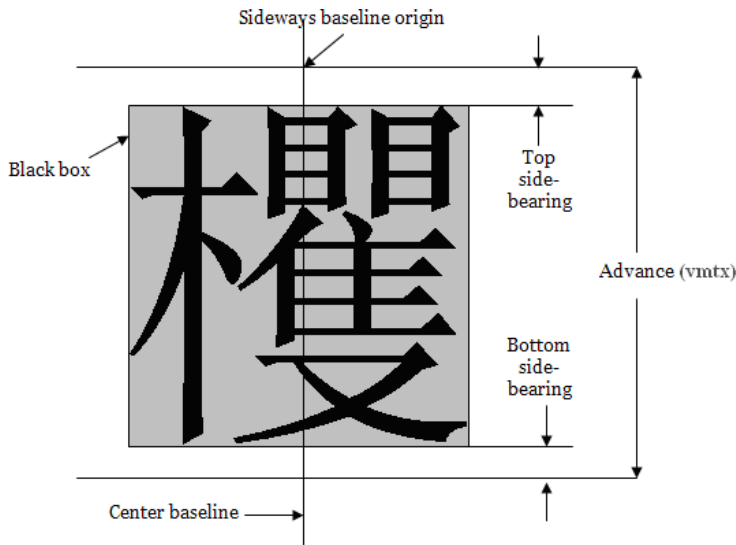


5 Black box origin

6 *Figure 12-2. Upright (usually horizontal) glyph metrics*



7

1 *Figure 12-3. Sideways (usually vertical) glyph metrics*

2

3 In general, glyphs within a font are either base glyphs or combining marks that can be attached
 4 to base glyphs. Base glyphs usually have an advance width that is non-zero, and a 0,0 offset
 5 vector. Combining marks usually have a zero advance width. The offset vector can be used to
 6 adjust the position of a combining mark and, therefore, can have a non-0,0 value for combining
 7 marks.

8 The position of each glyph in the glyph run is controlled by the following values:

- 9 • *Origin*. Each glyph is assumed to be given a nominal origin. For the first glyph in the run,
 10 this is the origin of the run.
- 11 • *Advance Width*. The advance width for each glyph provides the origin of the next glyph
 12 relative to the origin of the current glyph. The advance vector is drawn in the direction
 13 of the run progression.
- 14 • *Glyph Offset (Base or Mark)*. The glyph offset vector adjusts the position of this glyph
 15 relative to its nominal origin. The orientation of the glyph offset vector is not affected by
 16 the value of the *IsSideways* attribute, but is affected by the value of the *BidiLevel*
 17 attribute.

18 **12.1.2 Mapping Code Units to Glyphs**

19 A Unicode scalar value in a UnicodeString attribute is typically represented by a single UTF-16
 20 code unit and has a single corresponding glyph representation in the font. More complex
 21 mapping scenarios are common in non-Latin scripts: a single Unicode scalar value can map to
 22 two UTF-16 code units, multiple UTF-16 code units can map to a single glyph, single UTF-16
 23 code units can map to multiple glyphs based on context, and multiple UTF-16 code units can
 24 map indivisibly to multiple glyphs. In these cases, the clusters of UTF-16 code units are mapped
 25 using a cluster map.

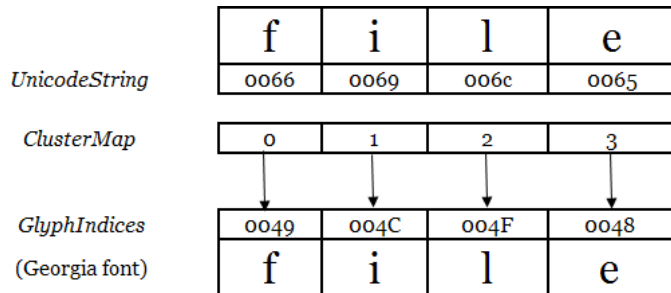
26 The cluster map contains one entry for each UTF-16 code unit in the UnicodeString attribute.
 27 Each entry specifies the offset of the first glyph that represents the cluster of UTF-16 code
 28 units.

1 **12.1.2.1 One-to-One Mappings**

2 When each UTF-16 code unit is represented by exactly one glyph, the cluster map entries are 0,
3 1, 2, and so on.

4 *Example 12-1. One-to-one cluster map*

5 Each character in the word "file" is represented by a single glyph.



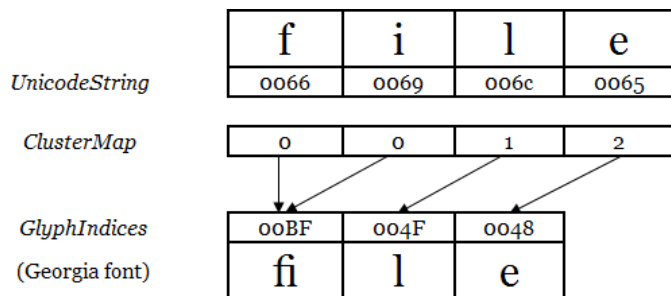
6
7 *end example]*

8 **12.1.2.2 Many-to-One Mappings**

9 When two or more UTF-16 code units map to a single glyph, the entries for those UTF-16 code
10 units specify the offset of that glyph in the glyph index buffer.

11 *Example 12-2. Many-to-one cluster map*

12 In the following mapping, the *f* and *i* characters are replaced by a ligature.



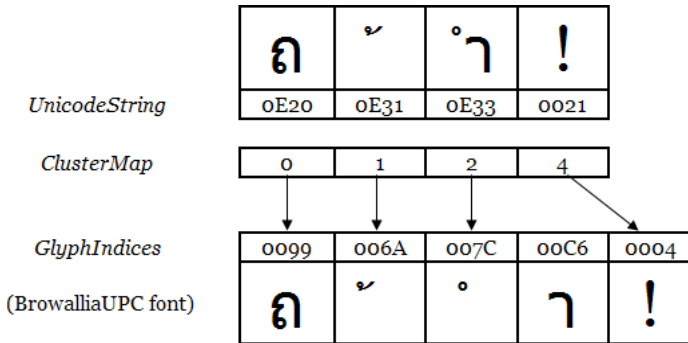
13
14 *end example]*

1 **12.1.2.3 One-to-Many Mappings**

2 When one UTF-16 code unit maps to two or more glyphs, the value in the cluster map for that
 3 UTF-16 code unit references the first glyph in the Indices attribute that represents that UTF-16
 4 code unit.

5 *Example 12-3. One-to-many cluster map*

6 The Thai *Sara Am* character contains a part that sits on top of the previous base character (the
 7 ring), and a part that sits to the right of the base character (the hook). When Thai text is
 8 micro-justified, the hook is spaced apart from the base character, while the ring remains on top
 9 of the base character. Many fonts encode the ring and the hook as separate glyphs.



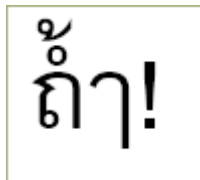
10

11 The markup appears as follows:

```

12 <Glyphs
13   FontUri="../Resources/Fonts/browau.ttf"
14   UnicodeString="&#xe20;&#xe3149;&#xe33;&#x21;"
15   Indices="153;106,,16;(1:2)124;198;4"
16   OriginX="10" OriginY="60"
17   FontRenderingEmSize="70"
18   Fill="#000000"/>
    
```

19 The markup above is rendered as follows:



20

21 *end example]*

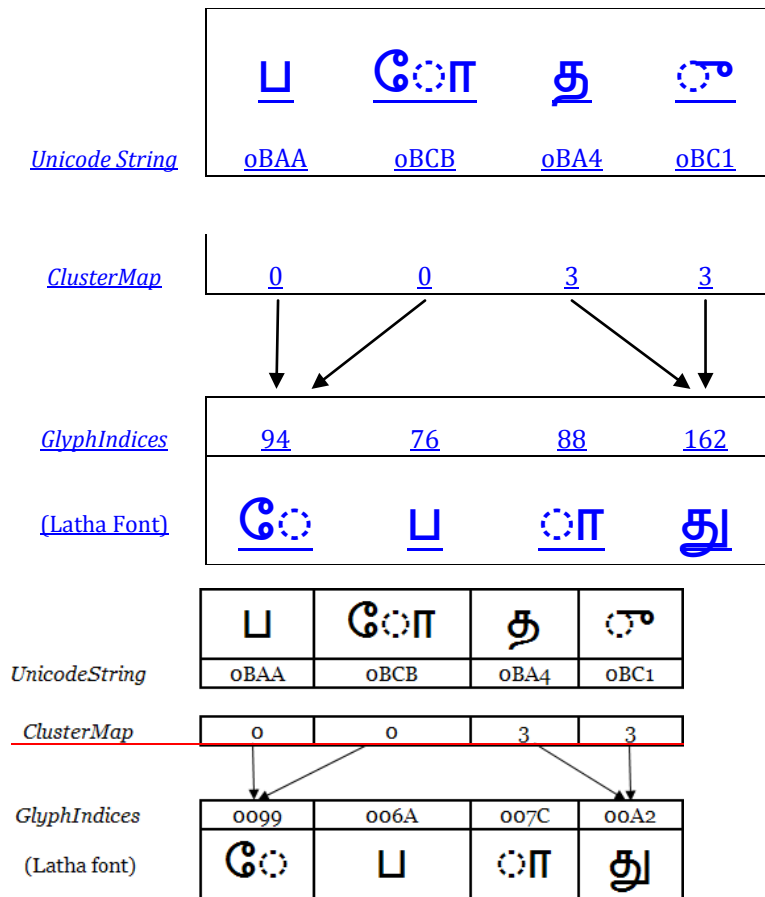
1 **12.1.2.4 Many-to-Many Mappings**

2 In some fonts, an indivisible group of UTF-16 code units for a character maps to more than one
 3 glyph. This is common in fonts that support Indic scripts. When an indivisible group of UTF-16
 4 code units maps to one or more glyphs, the value in the cluster map for each of the UTF-16
 5 code units references the first glyph in the Indices attribute representing that codepoint.

6 *Example 12-4. Many-to-many cluster map*

7 The following mapping shows the Unicode and glyph representations of a Tamil word that has
 8 two glyph clusters. Each cluster has a base character and a combining mark. The first pair of
 9 UTF-16 code units generates three glyphs because the combining mark splits both sides of the
 10 base character. The second pair of UTF-16 code units is represented by a single glyph that
 11 incorporates the effect of the combining mark.

12



13

14 The markup appears as follows:

```

15 <Glyphs
16   FontUri="../Resources/Fonts/latha.ttf"
17   UnicodeString="&#xbaa;&#x bcb;&#xba4;&#x bc1;"
18   Indices="(2:3)94;76;88;(2:1)162"
19   OriginX="10" OriginY="120"
20   FontRenderingEmSize="40"
21   Fill="#000000"/>
    
```

1 The markup above is rendered as follows:

A rectangular box with a thin black border containing the Tamil word 'போது' (Pothu) in a serif font. The word is centered within the box.

2

3 *end example]*

12.1.3 Indices Attribute

The <Glyphs> element MAY have an Indices attribute [M2.72]. The glyph specifications within the Indices attribute are OPTIONAL [M2.72]. The GlyphIndex portion of the Indices attribute MAY be used to specify a series of glyphs, complex character-to-glyph cluster mappings, or a combination of both [M2.72]. The Indices attribute MAY also include glyph placement information [M2.72].

Within the Indices attribute, each glyph specification is separated by a semicolon. The Indices attribute MUST adhere to the glyph specification syntax as follows [M2.72]:

```

9     GlyphIndices = *1GlyphMapping *( ";" *1GlyphMapping )
10    GlyphMapping = *1([ClusterMapping] GlyphIndex) [GlyphMetrics]
11    ClusterMapping = "(" ClusterCodeUnitCount ":" ClusterGlyphCount ")"
12    ClusterCodeUnitCount = 1*DIGIT
13    ClusterGlyphCount = 1*DIGIT
14    GlyphIndex = *DIGIT
15    GlyphMetrics = "," *1AdvanceWidth ["," *1uOffset ["," vOffset]]
16    AdvanceWidth = ["+"] RealNum
17    uOffset = ["+" | "-"] RealNum
18    vOffset = ["+" | "-"] RealNum
19    RealNum = ((1*DIGIT ["." 1*DIGIT]) | ( "." 1*DIGIT)) [Exponent]
20    Exponent = *1( ("E"|"e") ("+"|"-") 1*DIGIT )

```

The sum of the code unit counts for all the GlyphMapping entries in the Indices attribute MUST NOT exceed the number of UTF-16 code units in the UnicodeString attribute if the UnicodeString attribute is specified and does not contain an empty value ("{}"). If a ClusterMapping is not specified within a GlyphMapping entry, the code unit count is 1 [M5.4]. If the Indices attribute specifies a GlyphIndex that does not exist in the font, the consumer MUST generate an error [M5.24M5.4]. If the Indices attribute is specified, the values provided MUST be used in preference to values determined from the UnicodeString attribute alone [M5.23].

Table 12-3. Glyph specifications

Name	Description
GlyphIndex	<p>Index of the glyph (16-bit) in the physical font. The entry MAY be empty [M2.72], in which case the glyph index is determined by looking up the UTF-16 code unit in the font character map table. If there is not a one-to-one mapping between code units and the glyph indices, this entry MUST be specified [M5.5].</p> <p>In cases where character-to-glyph mappings are not one-to-one, a cluster mapping specification precedes the glyph index (further described below).</p>
AdvanceWidth	<p>Advance width indicating placement for the subsequent glyph, relative to the origin of the current glyph. Measured in direction of advance as defined by the IsSideways and BidiLevel attributes. Base glyphs generally have a non-zero advance width and combining glyphs have a zero advance width.</p> <p>Advance width is measured in hundredths of the font em size. The default value is defined in the horizontal metrics font table (hmtx) if the IsSideways attribute is specified as false or the vertical metrics font table (vmtx) if the IsSideways attribute is specified as true.</p> <p>Advance width is a real number with units specified in hundredths of</p>

an em.

So that rounding errors do not accumulate, the advance **MUST** be calculated as the exact unrounded origin of the subsequent glyph minus the sum of the calculated (that is, rounded) advance widths of the preceding glyphs [M5.6].

The advance **MUST** be 0 or greater [M2.72]. The right-to-left writing direction can be specified using the BidiLevel attribute.

uOffset, vOffset	<p>Offset in the effective coordinate space relative to glyph origin to move this glyph (x offset for uOffset and $-y$ offset for vOffset. The sign of vOffset is reversed from the direction of the y axis. A positive vOffset value shifts the glyph by a negative y offset and vice versa.). Used to attach marks to base characters. The value is added to the nominal glyph origin calculated using the advance width to generate the actual origin for the glyph. The setting of the IsSideways attribute does not change the interpretation of uOffset and vOffset.</p> <p>Measured in hundredths of the font em size. The default offset values are 0.0,0.0. uOffset and vOffset are real numbers.</p> <p>Base glyphs generally have a glyph offset of 0.0,0.0. Combining glyphs generally have an offset that places them correctly on top of the nearest preceding base glyph.</p> <p>For left-to-right text, a positive uOffset value points to the right; for right-to-left text, a positive uOffset value points to the left.</p>
------------------	--

1 *Example 12-5. Using indices to specify advance width*

2 The following Indices attribute specifies that the seventh glyph in the Unicode string has an
3 advance width of 40:

4 `Indices = ";;;;;;;;,40"`

5 *end example]*

6 **12.1.3.1 Specifying Character-to-Glyph Mappings**

7 A cluster map specification **MAY** precede the glyph specification for the first glyph of the cluster
8 [M2.72].

9 Empty Indices attribute values indicate that the corresponding UTF-16 code unit within the
10 Unicode string has a one-to-one relationship with the glyph index as specified by the character
11 mapping table within the font.

12 Cluster maps that specify 0:n or n:0 mappings are invalid.

13 See the glyph specification syntax above for details of how to specify cluster maps.

14 *Table 12-4. Portions of the cluster specification*

Name	Description
ClusterCodeUnitCount	Number of UTF-16 code units that combine to form this cluster. One or more code units can be specified. Default value is 1.
ClusterGlyphCount	Number of glyph indices that combine to form this cluster. One or more indices can be specified. Default value is 1.

1 *Example 12-6. Using the Indices attribute to specify glyph replacement for a cluster*

2 The following Indices attribute specifies that the sixth and seventh UTF-16 code units in the
3 Unicode string should be replaced by a single glyph having an index of 191:

4 `Indices = ";;;;(2:1)191"`

5 *end example]*

6 **12.1.4 UnicodeString Attribute**

7 The UnicodeString attribute holds the array of Unicode scalar values that are represented by the
8 current <Glyphs> element. Specifying a Unicode string is RECOMMENDED, as it supports
9 searching, selection, and accessibility [S5.5]. If the Unicode string contains Unicode scalar
10 values that require two UTF-16 code units, a cluster map with a many-to-one or many-to-many
11 mapping MUST be specified for the values [M5.5].

12 The standard XML escaping mechanisms are used to specify XML-reserved characters. An
13 additional mechanism MUST be used to escape a UnicodeString attribute value that begins with
14 an open brace (“{”) [M5.7].

15 In order to use an open brace at the beginning of the Unicode string, it MUST be escaped with a
16 prefix of “{” [M5.7]. If the UnicodeString attribute value starts with “{”, consumers MUST
17 ignore those first two characters in processing the Unicode string and in calculating index
18 positions for the characters of the Unicode string [M5.7].

19 If the UnicodeString attribute specifies an empty string (“” or “{}”), and the Indices attribute is
20 missing or is also empty, it MUST be treated as an error [M5.2]. If the UnicodeString attribute
21 contains a Unicode code unit that cannot be mapped to a glyph index via a cmap table in the
22 font and there is no corresponding GlyphIndex entry in the Indices attribute, the consumer
23 MUST display the .notdef glyph [M5.9].

24 Producers MAY include Unicode control marks in the Unicode string [O5.1]. Such marks include
25 control codes, layout controls, invisible operators, deprecated format characters, variation
26 selectors, non-characters, and specials, according to their definition within the Unicode
27 specification. If producers include control marks in the Unicode string, they SHOULD include an
28 Indices attribute to specify glyph indices and/or character-to-glyph mapping information for the
29 control marks [S5.2]. In the absence of such information, consumers MUST treat Unicode
30 control marks like ordinary characters and render the glyphs to which the Unicode control
31 marks are mapped in the CMAP table [M5.10]. The resulting glyphs might produce an
32 inappropriate rendering of the original Unicode string.

33 Producers MAY choose to generate UnicodeString attribute values that are not normalized by any
34 Unicode-defined algorithm [O5.2]. Because advance-widths, glyph indices, and caret-stops are
35 associated with the generated Unicode string, consumers MUST NOT normalize the UnicodeString
36 attribute value to produce an internal representation [M5.11]. See §9.1.7.5 for details and
37 exceptions.

38 **12.1.5 StyleSimulations Attribute**

39 Synthetic style simulations can be applied to the shape of the glyphs by using the
40 StyleSimulations attribute. Style simulations can be applied in addition to the designed style of a
41 font. The default value for the StyleSimulations attribute is None, in which case the shapes of
42 glyphs are not modified from their original design.

1 When the `StyleSimulations` value is specified as `BoldSimulation`, synthetic emboldening is applied
2 by geometrically widening the strokes of glyphs by 1% of the em size, so that the centers of
3 strokes remain at the same position. This leaves the baseline origin unmodified. The black box
4 grows 1% all around for a total of 2% horizontal and 2% vertical. As a result, the character
5 height and the advance width of each glyph are increased by 2% of the em size. Producers
6 MUST lay out algorithmically emboldened glyphs using advance widths that are 2% of the em
7 size larger than when not algorithmically emboldened [M5.12].

8 Consumers MUST implement the effect of algorithmic emboldening such that the black box of
9 the glyph grows by 2% of the em size [M5.13]. When advance widths are omitted from the
10 markup and the glyphs are algorithmically emboldened, the advance widths obtained from the
11 horizontal metrics font table (if `IsSideways` is false) or the vertical metrics font table (if
12 `IsSideways` is true) of the font MUST be increased by 2% of the em size [M5.13].

13 When `StyleSimulations` is specified as `ItalicSimulation`, synthetic italicizing is applied to glyphs
14 with an `IsSideways` value of false by skewing the top edge of the alignment box of the character
15 by 20° to the right, relative to the baseline of the character. Glyphs with an `IsSideways` value of
16 true are italicized by skewing the right edge of the alignment box of the character by 20° down,
17 relative to the baseline origin of the glyph. The character height and advance width are not
18 modified. Producers MUST lay out algorithmically italicized glyphs using exactly the same
19 advance widths as when not algorithmically italicized [M5.14].

20 When `StyleSimulations` is specified as `BoldItalicSimulation`, both `BoldSimulation` and
21 `ItalicSimulation` are applied, in order.

22 **12.1.6 IsSideways Attribute**

23 Glyphs for text in vertical writing systems are normally represented by rotating the coordinate
24 system and using the `IsSideways` attribute. `<Glyphs>` elements with the `IsSideways` attribute set
25 to true will be rotated 90° counter-clockwise and placed so that the sideways baseline origin is
26 coincident with the nominal origin of the character (within the glyph-local coordinate space), as
27 modified by the offset vector in the `Indices` attribute. The advance vector places the nominal
28 origin of the next character a distance along the direction of progression of the run. The
29 direction of the advance vector is unaffected by `IsSideways`, however the method by which the
30 size of the advance vector is chosen is different.

31 [*Example: To represent a run of characters top to bottom on a page, a render transform can be*
32 *used to rotate the <Glyphs> coordinate system 90° clockwise. OriginX and OriginY can be used*
33 *to specify a position at the top of the column of text. Text from a vertical writing system can*
34 *then be written using <Glyphs> elements with the IsSideways attribute set to true. The*
35 *individual glyphs appear in the normal orientation because the rotation effected by the*
36 *IsSideways attribute undoes the effect of the render transform. end example]*

37 Text from horizontal writing systems can be included in the column by using `<Glyphs>`
38 elements without specifying `IsSideways`, or using a value of false for it. The rotated coordinate
39 system makes them appear top to bottom on the page, but with the glyphs rotated to the right.

40 If alternate vertical character representations are available in the font, the producer SHOULD
41 use those and provide their glyph indices in the `Indices` attribute [S5.3].

42 **12.1.6.1 Calculating Sideways Text Origin and Advance Width**

43 The formulas below describe the method used to calculate each glyph's nominal origin, which is
44 used for positioning the glyphs on the fixed page and for calculating the default advance width
45 for each glyph.

1 The origin is the top center of the unturned glyph. The x origin of the unturned glyph is
 2 calculated to be exactly one-half the advance width of the glyph, as specified in the horizontal
 3 metrics table of the font. This formula is expressed as follows (in pseudocode):

4
$$\text{TopOriginX} = \text{hmtx.advanceWidth}[\text{GlyphIndex}] / 2$$

5 If the font is a CFF OpenType font, the y origin of the unturned glyph is determined from the
 6 vertical origin (vorg) table for the font, which can be specified for a particular glyph index but
 7 falls back to the default vertical origin if the glyph index is not present in the vertical origin
 8 table. This formula is expressed as follows (in pseudocode):

9
$$\text{TopOriginY} = \text{vorg.vertOriginY}[\text{glyphIndex}]$$

10 or:

11
$$\text{TopOriginY} = \text{vorg.defaultVertOriginY}$$

12 If the vertical origin table is not present, the glyph data (glyf) and vertical metrics (vmtx) font
 13 tables are consulted. The glyph bounding box is retrieved from the glyph data table and added
 14 to the top side-bearing for the glyph, specified in the vertical metrics table. This formula is
 15 expressed as follows (in pseudocode):

16
$$\text{TopOriginY} = \text{glyf.yMax}[\text{glyphIndex}] + \text{vmtx.topSideBearing}[\text{glyphIndex}]$$

17 [*Note: CFF fonts do not contain the glyf.yMax information; instead the yMax for each glyph is*
 18 *computed by calculating the top of the glyph's bounding box from the CFF charstring data. end*
 19 *note*]

20 If the vertical metrics font table does not exist but the Windows-specific metrics (OS/2) table
 21 does exist, the latter table is consulted and the sTypoAscender value is used. This formula is
 22 expressed as follows (in pseudocode):

23
$$\text{TopOriginY} = \text{os/2.sTypoAscender}$$

 24
$$\text{Descender} = \text{abs}(\text{os/2.typoDescender})$$

25 In all other circumstances, the Ascender value from the horizontal header (hhea) table is used.
 26 This formula is expressed as follows (in pseudocode):

27
$$\text{TopOriginY} = \text{hhea.Ascender}$$

 28
$$\text{Descender} = \text{abs}(\text{hhea.Descender})$$

29 Finally, the advance width for sideways text is computed as follows (in pseudocode), unless
 30 specifically overridden by the Indices attribute:

31
$$\text{AdvanceWidth} = \text{TopOriginY} + \text{Descender}$$

32 **12.1.6.2 IsSideways and BidiLevel Effects on Glyph Positioning**

33 Right-to-left text (BidiLevel attribute value of 1) changes the direction of the AdvanceWidth and
 34 uOffset (horizontal offset) values of the Indices attribute, as well as the position of the glyph
 35 origin. Vertical text (IsSideways attribute set to true) changes the position of the glyph origin.

36 Producers MUST NOT specify text that is both right-to-left (BidiLevel attribute value of 1) and
 37 vertical (IsSideways attribute set to true) [M5.15].

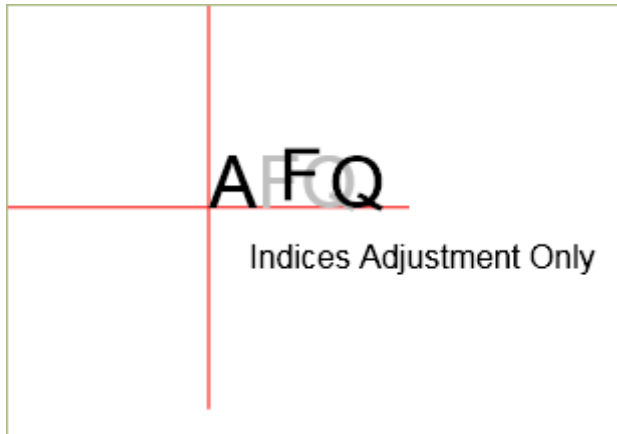
1 *Table 12–5. IsSideways and BidiLevel effects on origin placement*

IsSideways	BidiLevel	Glyph origin	Direction of advance width and positive uOffset
Horizontal (false)	Left-to-right (0)	Left end of horizontal advance vector along Latin baseline	To the right
Horizontal (false)	Right-to-left (1)	Right end of horizontal advance vector along Latin baseline	To the left
Vertical (true)	Left-to-right (0)	Top end of vertical advance vector through the glyph centerline	To the right
Vertical (true)	Right-to-left (1)	<i>Invalid combination</i>	

1 *Example 12-7. Text with positive uOffset and vOffset Indices values*

2 In this example, the position of the glyphs is shown relative to the origin shown at the crossed
3 lines centered at 100,100. The text in gray shows where this text would be rendered without
4 modification of the uOffset and vOffset value of the Indices attributes.

```
5 <Glyphs Fill="#000000" FontRenderingEmSize="48"  
6   OriginX="100" OriginY="100"  
7   UnicodeString="AFQ"  
8   Indices=";100,30,10;"  
9   FontUri=" ../Resources/Fonts/Arial.ttf" />
```



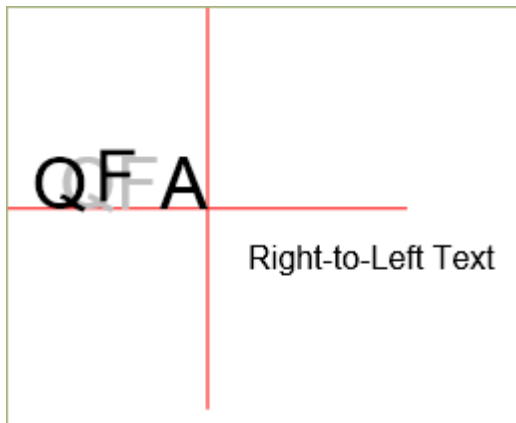
10

11 *end example]*

12 *Example 12-8. Right-to-left text (odd BidiLevel)*

13 The markup for this example matches the previous example, except the BidiLevel attribute is set
14 to 1. Note the change in the origin, and the reversal of the glyph advance direction.

```
15 <Glyphs Fill="#000000" FontRenderingEmSize="48"  
16   OriginX="100" OriginY="100"  
17   UnicodeString="AFQ"  
18   Indices=";100,30,10;"  
19   BidiLevel="1"  
20   FontUri=" ../Resources/Fonts/Arial.ttf" />
```



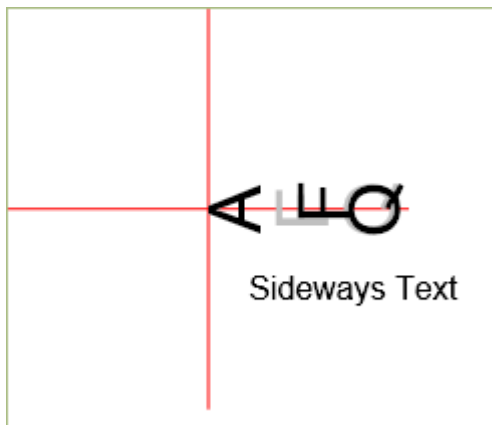
21

22 *end example]*

1 *Example 12-9. Sideways text (IsSideways set to true)*

2 This example shows the IsSideways attribute set to true. The BidiLevel ~~must~~**MUST** be even
 3 when the IsSideways attribute is set to true [M5.15]. Note that the origin has changed to be the
 4 top-center of the first glyph, with each glyph rotated 90° counter-clockwise. The interpretation
 5 of the advance direction and uOffset and vOffset values in the Indices attribute are otherwise
 6 unchanged.

```
7 <Glyphs Fill="#000000" FontRenderingEmSize="48"  
8   OriginX="100" OriginY="100"  
9   UnicodeString="AFQ"  
10  Indices=";,100,30,10;"  
11  IsSideways="true"  
12  FontUri="../Resources/Fonts/Arial.ttf" />
```



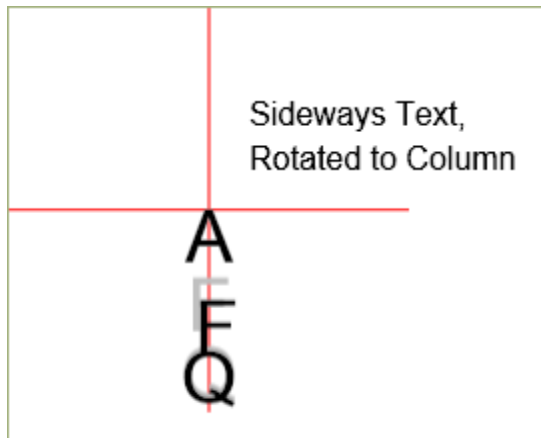
13

14 *end example]*

15 *Example 12-10. Vertical text*

16 The markup for this example matches the previous example, with the addition of a render
 17 transformation to rotate and position the element as vertical text. For more information on
 18 render transformations, see §14.4.

```
19 <Glyphs Fill="#000000" FontRenderingEmSize="48"  
20   OriginX="100" OriginY="100"  
21   UnicodeString="AFQ"  
22   Indices=";,100,30,10;"  
23   IsSideways="true"  
24   FontUri="../Resources/Fonts/Arial.ttf"  
25   RenderTransform="0,1,-1,0,200,0" />
```



1

2 *end example]*3 *Example 12-11. Japanese vertical text*

4 This example demonstrates a real-world usage of vertical text. Japanese text is shown below
 5 where the text is read down each column, from right to left across the page. The `IsSideways`
 6 attribute is set to `true`, thus rotating the each glyph 90° counter-clockwise. Then, the
 7 `RenderTransform` attribute (see §14.4) rotates the overall block of text 90° clockwise to achieve
 8 the final result of columns of text.

```

9     <Glyphs Fill="#000000" FontRenderingEmSize="24" OriginX="10" OriginY="10"
10         UnicodeString="これは、縦書きの日本語テキストが"
11         FontUri=" ../Resources/Fonts/msmincho.ttf" IsSideways="true"
12         RenderTransform="0,1,-1,0,145,0"/>
13     <Glyphs Fill="#000000" FontRenderingEmSize="24" OriginX="10" OriginY="45"
14         UnicodeString="どのように列で書かれるかの例です。"
15         FontUri=" ../Resources/Fonts/msmincho.ttf" IsSideways="true"
16         RenderTransform="0,1,-1,0,145,0"/>
17     <Glyphs Fill="#000000" FontRenderingEmSize="24" OriginX="10" OriginY="80"
18         UnicodeString="テキストは縦に読み、一行ずつ進みます。"
19         FontUri=" ../Resources/Fonts/msmincho.ttf" IsSideways="true"
20         RenderTransform="0,1,-1,0,145,0"/>
21     <Glyphs Fill="#000000" FontRenderingEmSize="24" OriginX="10"
22         OriginY="115" UnicodeString="他の言語も縦書きで書かれます。"
23         FontUri=" ../Resources/Fonts/msmincho.ttf" IsSideways="true"
24         RenderTransform="0,1,-1,0,145,0"/>

```

- 1 This markup is rendered as follows:

これは、縦書きの日本語テキストが
 どのように列で書かれるかの例です。
 テキストは縦に読み、一行ずつ進みます。
 他の言語も縦書きで書かれます。

2

3 *end example]*

4 **12.1.7 DeviceFontName Attribute**

5 Printer device fonts are specified by the DeviceFontName attribute. Device manufacturers define
 6 the values for this attribute. Producers SHOULD NOT produce markup that will result in different
 7 rendering between consumers using the embedded font to render and consumers using the
 8 device font to render [S5.4].

9 Consumers that understand the device font name MAY ignore the embedded font and use the
 10 device-resident version [O5.3]. By definition, a consumer “understands” a printer device font if
 11 it can unambiguously correlate the device font name to a set of font metrics resident on the
 12 device. If a consumer does not understand the specified device font name, it MUST render the
 13 embedded version of the font [M5.16].

14 When rendering a printer device font, consumers MUST use the UnicodeString attribute and
 15 ignore the glyph index components of the Indices attribute [M5.17]. The consumer MUST still
 16 honor the advance width and x,y offset values present in the Indices attribute [M5.18].

17 For producers, a <Glyphs> element with a specified device font name MUST have exactly one
 18 Indices glyph per code unit in the UnicodeString attribute. Its Indices attribute MUST NOT include
 19 any cluster specifications. If the Indices attribute includes a cluster mapping, the consumer
 20 MUST NOT use the device font and MUST render the embedded version of the font [M5.19].

1 This means that a device font cannot be used for characters outside the basic multilingual plane
2 (BMP).

3 If a device font name is specified, each of the <Glyphs> element's Indices glyphs MUST include
4 a specified advance width and MUST include specified x and y offset values if they are non-zero
5 [M5.20].

6 **12.1.8 xml:lang Attribute**

7 XPS Document consumers might need to override the default language for a specific run of
8 glyphs, particularly in multilingual documents. The language defaults to the value specified for
9 the xml:lang attribute of the <FixedPage> element but MAY be overridden by an xml:lang
10 attribute on a <Glyphs> element [M2.72]. For larger blocks of text, the producer MAY specify
11 the xml:lang attribute on the <Canvas> element [M2.72].

12 The language specified does not affect rendering of <Glyphs> elements, but it can be used by
13 consumers for searching or selecting text. For more information, see §9.3.5.

14 **12.1.9 CaretStops Attribute**

15 The CaretStops attribute contains an array of Boolean bit-flags, which is represented as a string
16 of hexadecimal characters. The flags indicate whether it is legal to place the caret before the
17 corresponding UTF-16 code unit in the UnicodeString attribute. ("Before" refers to a *logical*
18 placement, not a *physical* placement.) [*Example: If the flag is set in right-to-left text, the caret*
19 *can be placed before (to the right of) that UTF-16 code unit. end example*] The CaretStops
20 attribute includes a final flag for placement of the caret following the final UTF-16 code unit in
21 the Unicode string.

22 Each hexadecimal character in the CaretStops value represents the flags for four UTF-16 code
23 units in the Unicode string, with the highest-order bit representing the first UTF-16 code unit.
24 Any unused bits in the last UTF-16 code unit must be 0.

25 If the CaretStops attribute is omitted, it is legal to place the caret before any of the UTF-16 code
26 units in the Unicode string. Therefore, omitting the CaretStops attribute is equivalent to
27 specifying a string that has all the bits set to 1. If there are insufficient flags in the CaretStops
28 string to correspond to all the UTF-16 code units in the Unicode string, all remaining UTF-16
29 code units in the Unicode string MUST be considered valid caret stops [M5.22].

30 *Example 12-12. Using the CaretStops attribute to determine a valid caret stop position*

31 Given the following attributes, the *m* in "example" is not a valid caret stop position:

```
32 UnicodeString = "This is an example string of text."  
33 CaretStops = "fffd"
```

34 *end example*]

35 **12.1.10 Optimizing Glyph Markup**

36 Markup details such as glyph indices and advance widths can be omitted from the markup
37 under the circumstances described below. The following options allow optimization of commonly
38 used simple scripts.

39 **12.1.10.1 Optimizing Glyph Indices Markup**

40 Glyph indices MAY be omitted from markup where *all* of the following are true [O5.4]:

- 1 • There is a one-to-one mapping between the positions of Unicode scalar values in the
- 2 UnicodeString attribute and the positions of glyphs in the glyph string.
- 3 • The glyph index is the value in selected character mapping table of the font.

1 **12.1.10.2 Optimizing Glyph Position Markup**

2 Glyph advance width MAY be omitted from the markup in the following cases [O5.5]:

- 3 • For glyphs that have not been algorithmically emboldened, the desired advance width is
4 the value listed in the horizontal metrics font table (if the `IsSideways` attribute value is
5 false) or as calculated in §12.1.6.1 (if the `IsSideways` attribute value is true).
- 6 • For algorithmically emboldened glyphs, the desired advance width is exactly 2% larger
7 than the values in the horizontal metrics font table (if the `IsSideways` attribute value is
8 false) or as calculated in §12.1.6.1 (if the `IsSideways` attribute value is true).

9 Glyph horizontal offset MAY be omitted from the markup when the offset is 0.0, and Glyph
10 vertical offset MAY be omitted from the markup when the offset is 0.0 [O5.6]. This is almost
11 always true for base characters, and commonly true for combining marks in simple scripts.
12 However, this is often false for combining marks in complex scripts such as Arabic and Indic.

13 **12.1.11 Glyph Markup Examples**

14 *Example 12–13. Basic italic font*

```
15 <Canvas>
16 <Glyphs
17     FontUri=" ../Resources/Fonts/Timesi.ttf"
18     FontRenderingEmSize="20"
19     OriginX="35"
20     OriginY="35"
21     UnicodeString="Basic italic font..."
22     Fill="#009900" />
23 </Canvas>
```

24 This text is rendered as follows:



25 *Basic italic font...*

26 *end example]*

27 *Example 12–14. Italic font using StyleSimulations attribute*

```
28 <Canvas>
29 <Glyphs
30     FontUri=" ../Resources/Fonts/Times.ttf"
31     FontRenderingEmSize="20"
32     StyleSimulations="ItalicSimulation"
33     OriginX="35"
34     OriginY="35"
35     UnicodeString="Simulated italic font..."
36     Fill="#009900" />
37 </Canvas>
```

38 This text is rendered as follows:

Simulated italic font...

1

2 *end example]*3 *Example 12-15. Kerning*

4 <Canvas>

5

6 <!-- "WAVE" without kerning -->

7

8 <Glyphs

9 OriginX="35"

10 OriginY="35"

11 UnicodeString="WAVE (no kerning)"

12 FontUri="../Resources/Fonts/Times.ttf"

13 FontRenderingEmSize="20"

14 Fill="#009900" />

15

16 <!-- "WAVE" with kerning -->

17

18 <Glyphs

19 OriginX="35"

20 OriginY="70"

21 UnicodeString="WAVE (with kerning)"

22 Indices=",88;,59"

23 FontUri="../Resources/Fonts/Times.ttf"

24 FontRenderingEmSize="20"

25 Fill="#009900" />

26

27 </Canvas>

28 This text is rendered as follows:

WAVE (no kerning)

WAVE (with kerning)

29

30 *end example]*31 *Example 12-16. Ligatures*

32 <Canvas>

33

34 <!-- "Open file" without "fi" ligature -->

35

36 <Glyphs

37 OriginX="35"

38 OriginY="35"

39 UnicodeString="Open file (no ligature)"

40 FontUri="../Resources/Fonts/Times.ttf"

41 FontRenderingEmSize="20"

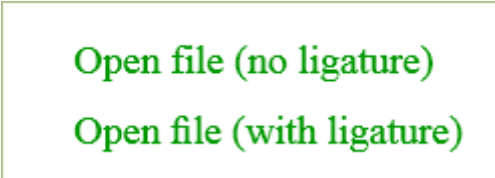
42 Fill="#009900" />

```

1
2     <!-- "Open file" with "fi" ligature -->
3
4     <Glyphs
5         OriginX="35"
6         OriginY="70"
7         UnicodeString="Open file (with ligature)"
8         Indices=";;;;(2:1)191"
9         FontUri="../Resources/Fonts/Times.ttf"
10        FontRenderingEmSize="20"
11        Fill="#009900" />
12
13 </Canvas>

```

14 This text is rendered as follows:



15

16 *end example]*

17 *Example 12-17. Cluster maps*

```

18     <Canvas>
19
20     <!-- "ёжик в тумане" using pre-composed "ё" -->
21
22     <Glyphs
23         OriginX="35"
24         OriginY="35"
25         xml:lang="ru-RU"
26         UnicodeString="ёжик в тумане"
27         FontUri="../Resources/Fonts/Times.ttf"
28         FontRenderingEmSize="20"
29         Fill="#009900" />
30
31     <!-- "ёжик в тумане" using composition of "e" and diaeresis -->
32
33     <Glyphs
34         OriginX="35"
35         OriginY="70"
36         xml:lang="ru-RU"
37         UnicodeString="ёжик в тумане"
38         Indices="(1:2)72;142,0,-40"
39         FontUri="../Resources/Fonts/Times.ttf"
40         FontRenderingEmSize="20"
41         Fill="#009900" />
42
43     <!-- "ёжик в тумане" Forced rendering right-to-left showing
44     combining mark in logical order -->
45
46     <Glyphs

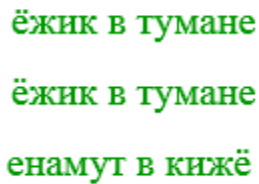
```

```

1      OriginX="155"
2      OriginY="105"
3      BidiLevel="1"
4      xml:lang="ru-RU"
5      UnicodeString="ёжик в тумане"
6      Indices="(1:2)72;142,0,-40"
7      FontUri=" ../Resources/Fonts/Times.ttf"
8      FontRenderingEmSize="20"
9      Fill="#009900" />
10
11 </Canvas>

```

12 This text is rendered as follows:

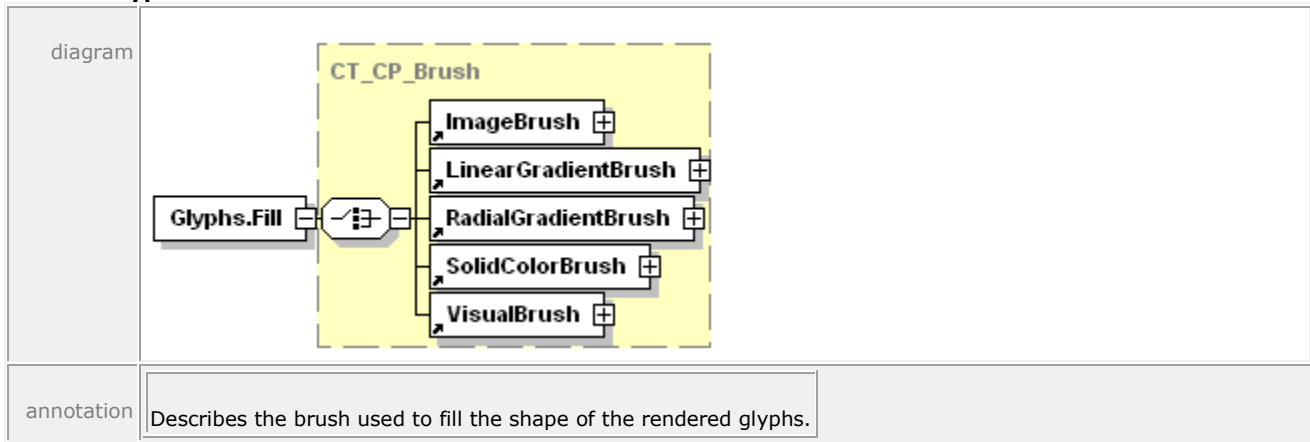


ёжик в тумане
ёжик в тумане
енамут в кижё

13
14 *end example]*

15 12.2 <Glyphs.Fill> Element

16 element **Glyphs.Fill**



17 The Fill property specifies the brush that fills a glyph. Any brush can be used.

18

1 13. Brushes

2 Brushes are used to paint the interior of the geometric shapes defined by a <Path> element
 3 and the characters rendered with a <Glyphs> element. They are also used to define the alpha-
 4 transparency mask in the <Canvas.OpacityMask>, <Path.OpacityMask>, and
 5 <Glyphs.OpacityMask> property elements.

6 All brushes are defined relative to a coordinate space. Most brushes (including image brushes,
 7 visual brushes, linear gradient brushes, and radial gradient brushes) can specify a coordinate-
 8 space transform, in which the Transform property is concatenated with the current effective
 9 coordinate space to yield an effective coordinate space local to the brush. For image brushes
 10 and visual brushes, the viewport is transformed using the local effective render transform. For
 11 linear gradient brushes, the start point and end point are transformed. For radial gradient
 12 brushes, the ellipse defined by the center, x radius, y radius, and gradient origin is
 13 transformed.

14 *Table 13-1. Brush types*

Name	Description
Solid color brush	Fills a region with a solid color
Image brush	Fills a region with an image
Visual brush	Fills a region with a drawing
Linear gradient brush	Fills a region with a linear gradient
Radial gradient brush	Fills a region with a radial gradient

1 **13.1 <SolidColorBrush> Element**

2 element **SolidColorBrush**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.1].
	Color	<u>ST_Color</u>	required			Specifies the color for filled elements. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.
annotation	Fills defined geometric regions with a solid color.					

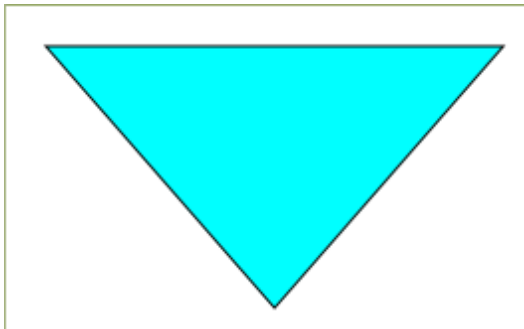
3 The <SolidColorBrush> element is used to fill defined geometric regions with a solid color. If
 4 there is an alpha component of the color, it is combined in a multiplicative way with the
 5 corresponding Opacity attribute.

1 *Example 13-1. <SolidColorBrush> usage*

2 The following markup illustrates how a solid color brush fills a path.

```
3     <Path Stroke="#000000">
4         <Path.Fill>
5             <SolidColorBrush Color="#00FFFF" />
6         </Path.Fill>
7         <Path.Data>
8             <PathGeometry>
9                 <PathFigure StartPoint="20,20" IsClosed="true">
10                    <PolyLineSegment Points="250,20 135,150" />
11                </PathFigure>
12            </PathGeometry>
13        </Path.Data>
14    </Path>
```

15 This markup is rendered as follows:

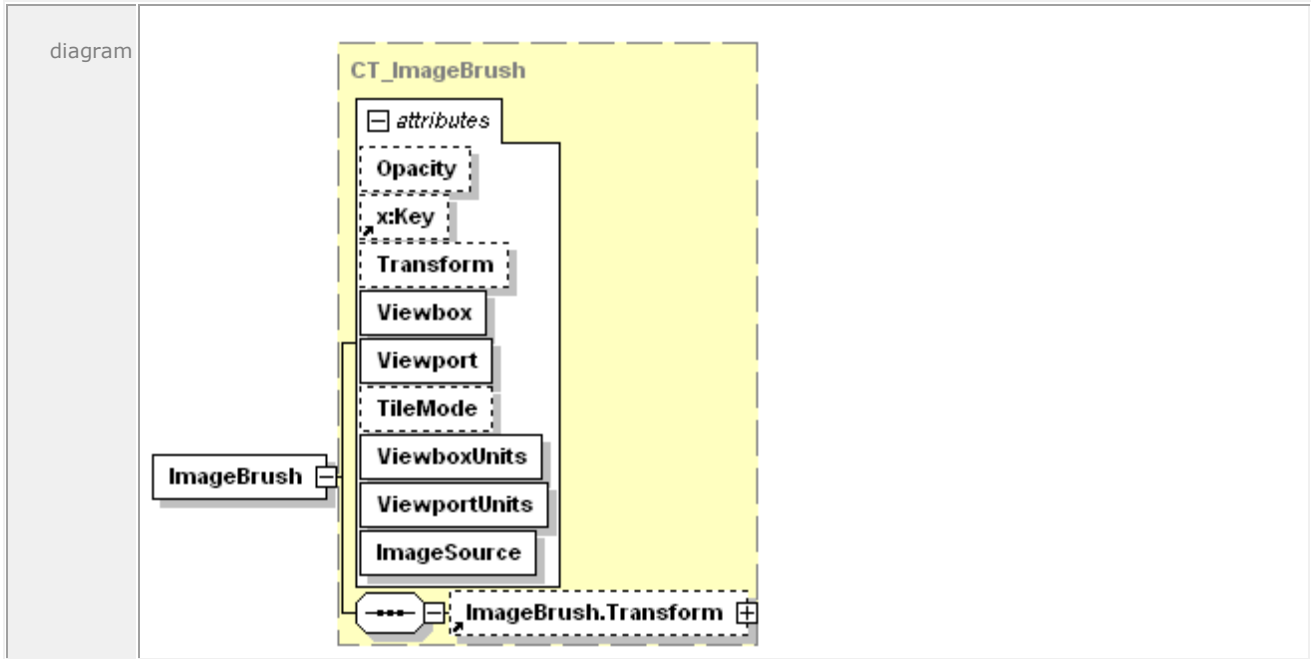


16

17 *end example]*

18 **13.2 <ImageBrush> Element**

19 element **ImageBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.2].
	Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.
	Viewbox	ST_ViewBox	required			Specifies the position and dimensions of the brush's source content. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The dimensions specified are relative to the image's physical dimensions expressed in units of 1/96". The corners of the viewbox are mapped to the corners of the viewport, thereby providing the default clipping and transform for the brush's source content.

	Viewport	ST_ViewBox	required			Specifies the region in the containing coordinate space of the prime brush tile that is (possibly repeatedly) applied to fill the region to which the brush is applied. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The alignment of the brush pattern is controlled by adjusting the x and y values.
	TileMode	ST_TileMode		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
	ViewboxUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
	ViewportUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
	ImageSource	ST_UriCtxBmp	required			Specifies the URI of an image resource or a combination of the URI of an image resource a color profile resource. See the Color clause for important details. The URI MUST refer to parts in the package [M2.1].
annotation	Fills a region with an image.					

1 The <ImageBrush> element is used to fill a region with an image. The image is defined in a
2 coordinate space specified by the resolution of the image. The image MUST refer to a JPEG,
3 PNG, TIFF-6.0, or Windows Media Photo image part within the XPS Document package [M6.3].
4 For more information, see §9.1.5. A URI part name for the image is specified using the
5 ImageSource attribute.

6 Image brushes share a number of tile-related properties with visual brushes. For details,
7 see §13.4.

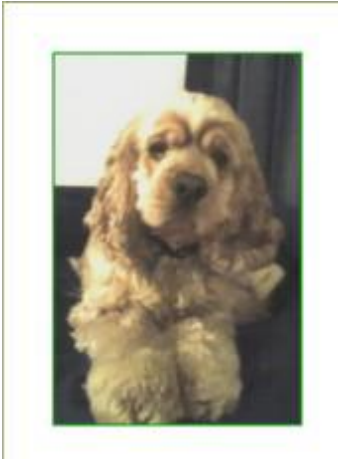
8 *Example 13-2. <ImageBrush> usage*

9 The following markup describes an image on a canvas.

```
10 <Canvas>
11   <Path Stroke="#008000">
12     <Path.Fill>
13       <ImageBrush
14         ImageSource="dog.jpg"
15         TileMode="None"
16         Viewbox="0,0,270,423"
17         ViewboxUnits="Absolute"
18         Viewport="25,25,125,185"
```

```
1         ViewportUnits="Absolute" />
2     </Path.Fill>
3     <Path.Data>
4         <PathGeometry>
5             <PathFigure StartPoint="25,25" IsClosed="true">
6                 <PolyLineSegment Points="150,25 150,210 25,210" />
7             </PathFigure>
8         </PathGeometry>
9     </Path.Data>
10 </Path>
11 </Canvas>
```

12 This markup produces the following results:

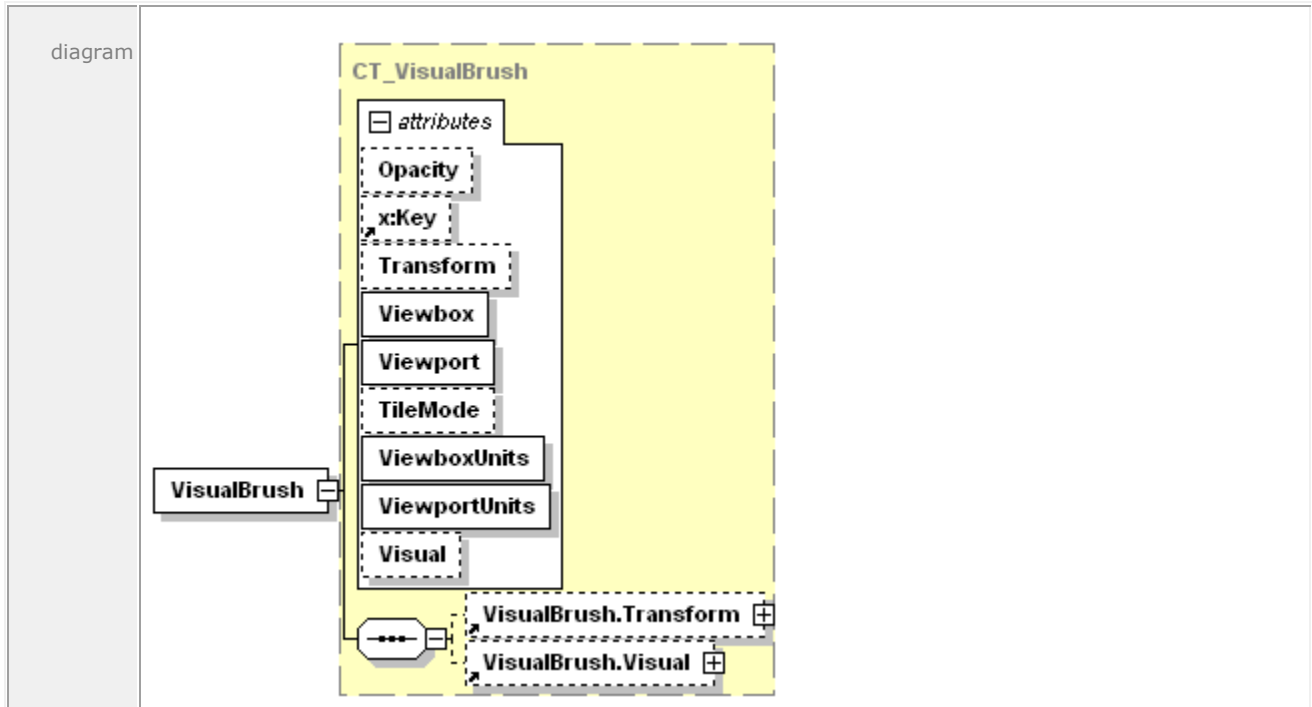


13

14 *end example]*

1 **13.3 <VisualBrush> Element**

2 element **VisualBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.4].
	Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using that local effective render transform.
	Viewbox	ST_ViewBox	required			Specifies the position and dimensions of the brush's source content. Specifies four comma-separated real numbers (x, y, Width, Height),

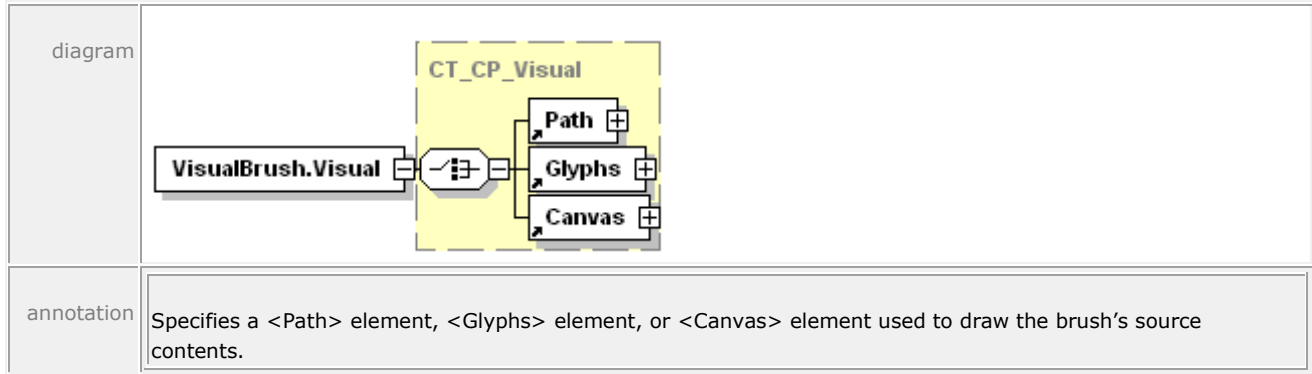
					where width and height are non-negative. The viewbox defines the default coordinate system for the element specified in the <VisualBrush.Visual> property element. The corners of the viewbox are mapped to the corners of the viewport, thereby providing the default clipping and transform for the brush's source content.
Viewport	ST_ViewBox	required			Specifies the region in the containing coordinate space of the prime brush tile that is (possibly repeatedly) applied to fill the region to which the brush is applied. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The alignment of the brush pattern is controlled by adjusting the x and y values.
TileMode	ST_TileMode		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
Visual	ST_RscRef				Specifies resource reference to a <Path>, <Glyphs>, or <Canvas> element defined in a resource dictionary and used to draw the brush's source content.
annotation	Fills a region with a drawing. The drawing can be specified as either a child of the <VisualBrush> element, or as a resource reference. Drawing content is expressed using <Canvas>, <Path>, and <Glyphs> elements.				

1 The <VisualBrush> element is used to fill a region with a drawing. The drawing can be specified
 2 as either a <VisualBrush.Visual> property element or as a resource reference. Drawing content
 3 can include exactly one <Canvas>, <Path>, or <Glyphs> element and that element's child and
 4 descendant elements.

5 Visual brushes share a number of tile-related properties with image brushes. For details,
 6 see §13.4.

7 **13.3.1 <VisualBrush.Visual> Element**

8 element **VisualBrush.Visual**



- 1 The <VisualBrush.Visual> property element contains markup that defines the contents of a
- 2 single visual brush tile. The tile can be used to fill the geometric region to which the visual
- 3 brush is applied. The <VisualBrush.Visual> property element contains a single child element.
- 4 For simple tiles, this can be a single <Path> or <Glyphs> element. More complex visuals
- 5 containing multiple <Path> and <Glyphs> elements can be grouped within a <Canvas> child
- 6 element.

1 *Example 13-3. <VisualBrush.Visual> usage*

```
2     <Path>
3         <Path.Fill>
4             <VisualBrush
5                 Viewbox="0,0,1,1"
6                 Viewport="50,50,100,100"
7                 ViewportUnits="Absolute"
8                 ViewboxUnits="Absolute"
9                 TileMode="Tile">
10                <VisualBrush.Visual>
11                    <Path>
12                        <Path.Fill>
13                            <SolidColorBrush Color="#FF0000" />
14                        </Path.Fill>
15                        <Path.Data>
16                            <PathGeometry>
17                                <PathFigure StartPoint="0,0.5" IsClosed="true">
18                                    <PolyLineSegment Points="0.5,0 1.0,0.5
19                                        0.5,1.0" />
20                                </PathFigure>
21                            </PathGeometry>
22                        </Path.Data>
23                    </Path>
24                </VisualBrush.Visual>
25            </VisualBrush>
26        </Path.Fill>
27        <Path.Data>
28            <PathGeometry>
29                <PathFigure StartPoint="50,50" IsClosed="true">
30                    <PolyLineSegment Points="350,50 350,350 50,350" />
31                </PathFigure>
32            </PathGeometry>
33        </Path.Data>
34    </Path>
```


1 This markup produces the following result:



2

3 *end example]*

13.4 Common Attributes for Tiling Brushes

Image brushes and Visual brushes share certain tiling characteristics in common. These characteristics are controlled by a common set of attributes described in the table below.

Table 13–2. Common attributes for <ImageBrush> and <VisualBrush> elements

Name	Description
Viewbox	Specifies the region of the source content of the brush that is to be mapped to the viewport.
Viewport	Specifies the position and dimensions of the first brush tile. Subsequent tiles are positioned relative to this tile, as specified by the tile mode.
ViewboxUnits	Specifies the unit type for the Viewbox attribute. MUST have the value "Absolute" [M2.72].
ViewportUnits	Specifies the unit type for the Viewport attribute. MUST have the value "Absolute" [M2.72].
TileMode	Specifies how tiling is performed in the filled geometry. The value is optional, and defaults to "None" if no value is specified.

Both image brushes and visual brushes assume that the background of the brush itself is initially transparent.

13.4.1 Viewbox, Viewport, ViewboxUnits, and ViewportUnits Attributes

The Viewbox attribute specifies the portion of a source image or visual to be rendered to the page as a tile. The Viewport attribute specifies the dimensions and location, in the effective coordinate space, of the initial tile that will be filled with the specified image or visual fragment. In other words, the Viewport attribute defines the initial tile whose origin (x and y values of the top left corner of the tile relative to the current effective render transform) is specified by the first two parameters and whose size (width and height values) is specified by the last two parameters. The tile is then used to fill the geometry specified by the parent element according to the TileMode attribute relative to the initial tile.

For images, the dimensions specified by the viewbox are expressed in units of 1/96". The pixel coordinates in the source image are calculated as follows:

```
SourceLeft = HorizontalImageResolution * Viewbox.Left / 96
SourceTop = VerticalImageResolution * Viewbox.Top / 96
SourceWidth = HorizontalImageResolution * Viewbox.Width / 96
SourceHeight = VerticalImageResolution * Viewbox.Height / 96
```

The image resolution used is that specified in the header or tag information of the image. If no resolution is specified, a default resolution of 96 dpi is assumed. The coordinates of the upper-left corner of the image are 0,0.

The viewbox can specify a region larger than the image itself, including negative values.

1 *Example 13-4. ViewboxUnits and ViewportUnits attribute usage*

2 The following markup contains an image brush:

```
3 <ImageBrush
4   ImageSource=" ../Resources/Images/tiger.jpg"
5   Viewbox="24,24,48,48"
6   ViewboxUnits="Absolute"
7   Viewport="96,96,192,192"
8   ViewportUnits="Absolute"
9   TileMode="None" />
```

10 Assuming the default fixed page coordinate system and that tiger.jpg specifies a resolution of
11 50 dpi and measures 100 pixels horizontally and 50 pixels vertically, the physical dimensions of
12 the image are expressed (in units of 1/96") as $96 * 100 / 50 = 192$ horizontal and $96 * 50 / 50$
13 $= 96$ vertical.

14 The viewbox uses a square starting at 24,24 (a quarter-inch from left and a quarter-inch from
15 top) in the image, and extending for 48,48 (a half-inch to the right and a half-inch down) and
16 scales it to a square starting at one inch from the left edge of the physical page and one inch
17 from the top of the physical page and extending two inches to the right and two inches down.
18 *end example]*

19 **13.4.1.1 Viewbox and Viewport Examples**

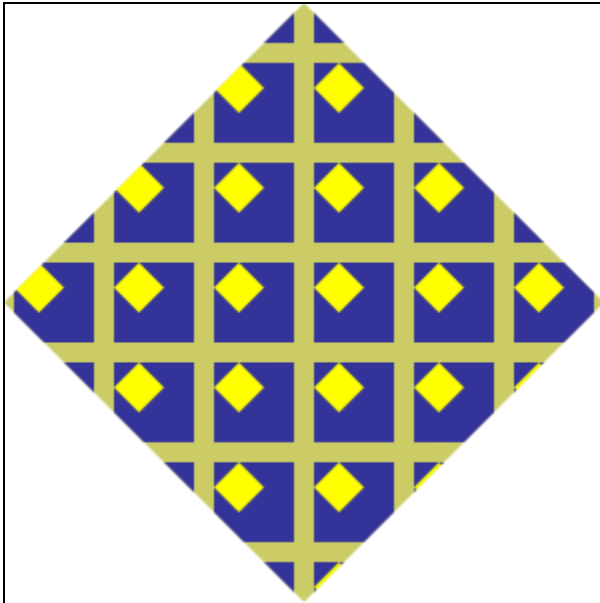
20 The following examples demonstrate how adjusting the viewbox and viewport can affect output.

21 *Example 13-5. Tiling brush base image and rendering*

22 The following markup describes a base image.

```
23 <!-- Draw background diamond to show where fill affects background -->
24 <Path Fill="#CCCC66" Data="M 150,0 L 300,150 L 150,300 L 0,150 Z" />
25 <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
26   <Path.Fill>
27     <VisualBrush
28       Viewbox="0,0,1,1"
29       Viewport="150,75,50,50"
30       ViewboxUnits="Absolute"
31       ViewportUnits="Absolute"
32       TileMode="Tile">
33       <VisualBrush.Visual>
34         <Canvas>
35           <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
36             L 0.1,0.9 Z" />
37           <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
38             L 0.6,0.35 L 0.35,0.6 Z" />
39         </Canvas>
40       </VisualBrush.Visual>
41     </VisualBrush>
42   </Path.Fill>
43 </Path>
```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13-6. Tiling brush Viewport adjustments*

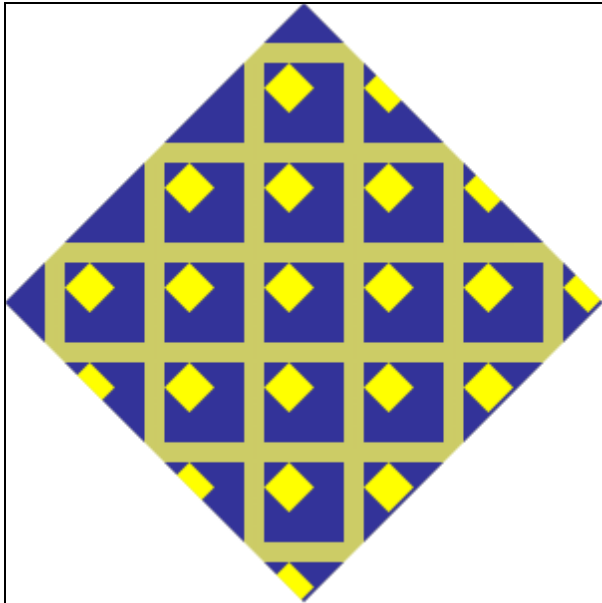
5 By adjusting the viewport, the position of the tiles within the image can be changed:

```

6 <!-- Draw background diamond to show where fill affects background -->
7 <Path Fill="#CCCC66" Data="M 150,0 L 300,150 L 150,300 L 0,150 Z" />
8 <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
9   <Path.Fill>
10     <VisualBrush
11       Viewbox="0,0,1,1"
12       Viewport="125,125,50,50"
13       ViewboxUnits="Absolute"
14       ViewportUnits="Absolute"
15       TileMode="Tile">
16       <VisualBrush.Visual>
17         <Canvas>
18           <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
19             L 0.1,0.9 Z" />
20           <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
21             L 0.6,0.35 L 0.35,0.6 Z" />
22         </Canvas>
23       </VisualBrush.Visual>
24     </VisualBrush>
25   </Path.Fill>
26 </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13-7. Tiling brush viewbox adjustments*

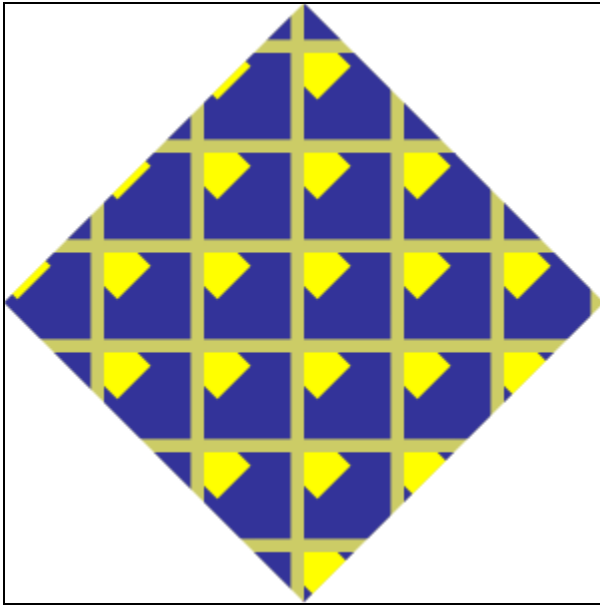
5 The following markup uses a smaller window on the viewbox to zoom in on each tile:

```

6 <!-- Draw background diamond to show where fill affects background -->
7 <Path Fill="#CCCC66" Data="M 150,0 L 300,150 L 150,300 L 0,150 Z" />
8 <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
9   <Path.Fill>
10     <VisualBrush
11       Viewbox="0.25,0.25,0.75,0.75"
12       Viewport="150,75,50,50"
13       ViewboxUnits="Absolute"
14       ViewportUnits="Absolute"
15       TileMode="Tile">
16       <VisualBrush.Visual>
17         <Canvas>
18           <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
19             L 0.1,0.9 Z" />
20           <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
21             L 0.6,0.35 L 0.35,0.6 Z" />
22         </Canvas>
23       </VisualBrush.Visual>
24     </VisualBrush>
25   </Path.Fill>
26 </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13-8. Image brush with a Viewbox larger than the image*

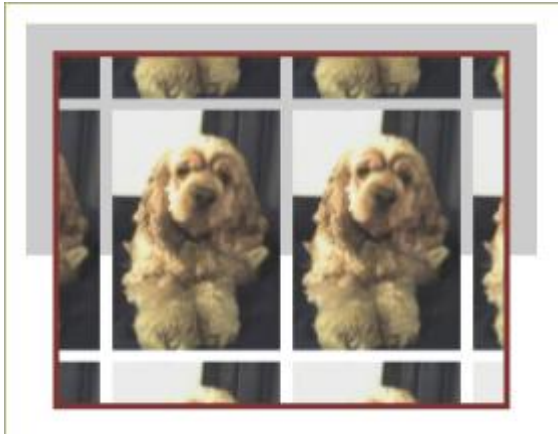
5 An image brush can specify a tile with the Viewbox attribute that exceeds the size of the image
6 it uses, including negative values, as shown below.

```

7     <Path Fill="#CCCCCC" Data="M 10,10 L 265,10 L 265,125 L 10,125 Z" />
8     <Path Stroke="#803333" StrokeThickness="3"
9         Data="M 25,25 L 250,25 L 250,200 L 25,200 Z">
10        <Path.Fill>
11            <ImageBrush ImageSource="../Resources/Images/dog.jpg"
12                TileMode="Tile"
13                Viewbox="-10,-10,290,443" ViewboxUnits="Absolute"
14                Viewport="50,50,90,125" ViewportUnits="Absolute" />
15        </Path.Fill>
16    </Path>

```

- 1 This markup is rendered as follows. Note that the area around the image is transparent,
 2 revealing the underlying path between the tiles.



- 3
 4 *end example]*

5 **13.4.2 TileMode Attribute**

- 6 Valid values for the TileMode attribute are None, Tile, FlipX, FlipY, and FlipXY.

7 **13.4.2.1 None**

- 8 In this mode, only the single base tile is drawn. The remaining area is left transparent.

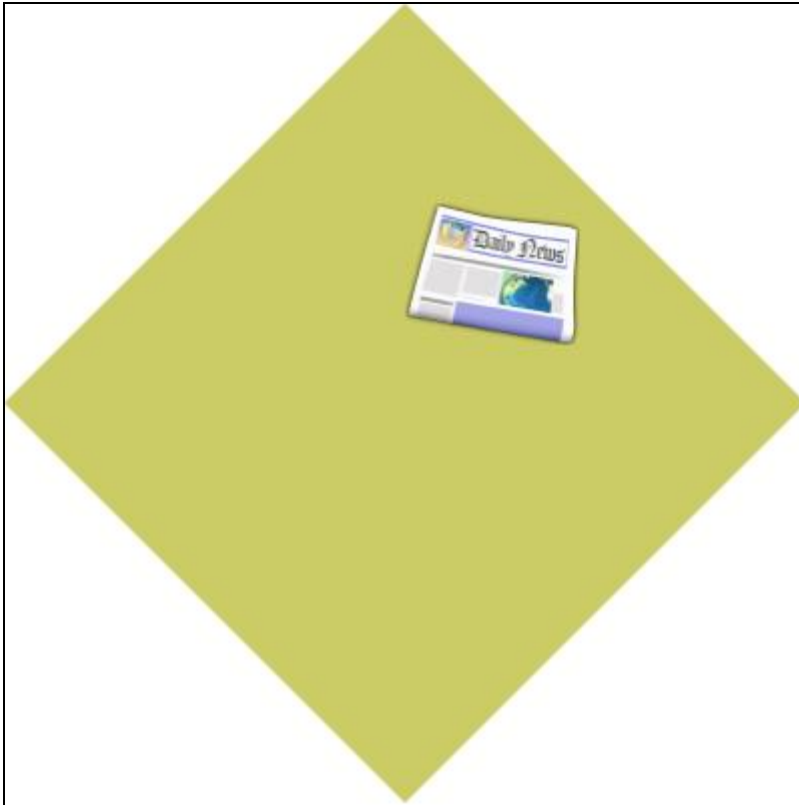
9 *Example 13-9. Image brush with TileMode value of None*

```

10 <!-- Draw background diamond to show where fill affects background -->
11 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
12 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
13   <Path.Fill>
14     <ImageBrush
15       ImageSource="newspaper.png"
16       Viewbox="0,0,350,284"
17       Viewport="200,100,87,71"
18       ViewportUnits="Absolute"
19       ViewboxUnits="Absolute"
20       TileMode="None" />
21   </Path.Fill>
22 </Path>

```

1 This markup is rendered as follows:



2

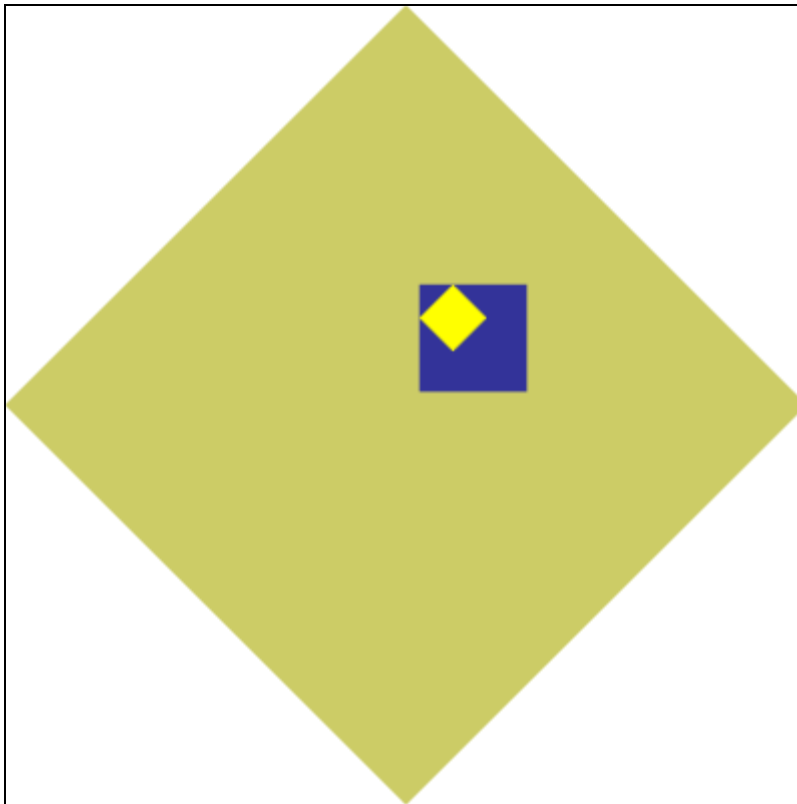
3 *end example]*


```

1  Example 13-10. Visual brush with TileMode value of None
2  <!-- Draw background diamond to show where fill affects background -->
3  <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
4  <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
5      <Path.Fill>
6          <VisualBrush
7              Viewbox="0,0,1,1"
8              Viewport="200,133,67,67"
9              ViewboxUnits="Absolute"
10             ViewportUnits="Absolute"
11             TileMode="None">
12                 <VisualBrush.Visual>
13                     <Canvas>
14                         <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
15                             L 0.1,0.9 Z" />
16                         <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
17                             L 0.6,0.35 L 0.35,0.6 Z" />
18                     </Canvas>
19                 </VisualBrush.Visual>
20             </VisualBrush>
21         </Path.Fill>
22     </Path>

```

23 This markup is rendered as follows:



24

25 *end example]*

1 **13.4.2.2 Tile**

2 In this mode, the base tile is drawn and the remaining area is filled by repeating the base tile
3 such that the right edge of each tile abuts the left edge of the next, and the bottom edge of
4 each tile abuts the top edge of the next.

5 *Example 13–11. Image brush with a TileMode value of Tile*

```
6 <!-- Draw background diamond to show where fill affects background -->  
7 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />  
8 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">  
9 <Path.Fill>  
10 <ImageBrush  
11 <ImageSource="newspaper.png"  
12 <Viewbox="0,0,350,284"  
13 <Viewport="200,100,87,71"  
14 <ViewportUnits="Absolute"  
15 <ViewboxUnits="Absolute"  
16 <TileMode="Tile" />  
17 </Path.Fill>  
18 </Path>
```

19 This markup is rendered as follows:



20

21 *end example]*

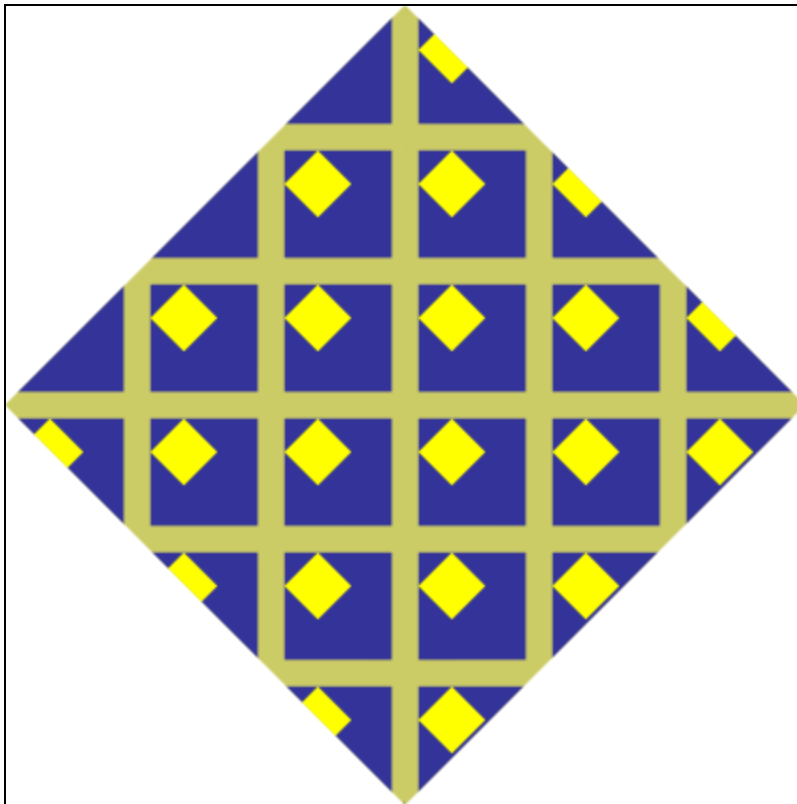
1 *Example 13-12. Visual brush with a TileMode value of Tile*

```

2 <!-- Draw background diamond to show where fill affects background -->
3 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
4 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
5   <Path.Fill>
6     <VisualBrush
7       Viewbox="0,0,1,1"
8       Viewport="200,133,67,67"
9       ViewboxUnits="Absolute"
10      ViewportUnits="Absolute"
11      TileMode="Tile">
12        <VisualBrush.Visual>
13          <Canvas>
14            <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
15              L 0.1,0.9 Z" />
16            <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
17              L 0.6,0.35 L 0.35,0.6 Z" />
18          </Canvas>
19        </VisualBrush.Visual>
20      </VisualBrush>
21    </Path.Fill>
22  </Path>

```

23 This markup is rendered as follows:



24

25 *end example]*

1 **13.4.2.3 FlipX**

2 The tile arrangement is similar to the Tile tile mode, but alternate columns of tiles are flipped
3 horizontally. The base tile is positioned as specified by the viewport. Tiles in the columns to the
4 left and right of this tile are flipped horizontally.

5 *Example 13–13. Image brush with a TileMode value of FlipX*

```
6 <!-- Draw background diamond to show where fill affects background -->  
7 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />  
8 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">  
9 <Path.Fill>  
10 <ImageBrush  
11 <ImageSource="newspaper.png"  
12 <Viewbox="0,0,350,284"  
13 <Viewport="200,100,87,71"  
14 <ViewportUnits="Absolute"  
15 <ViewboxUnits="Absolute"  
16 <TileMode="FlipX" />  
17 </Path.Fill>  
18 </Path>
```

19 This markup is rendered as follows:



20

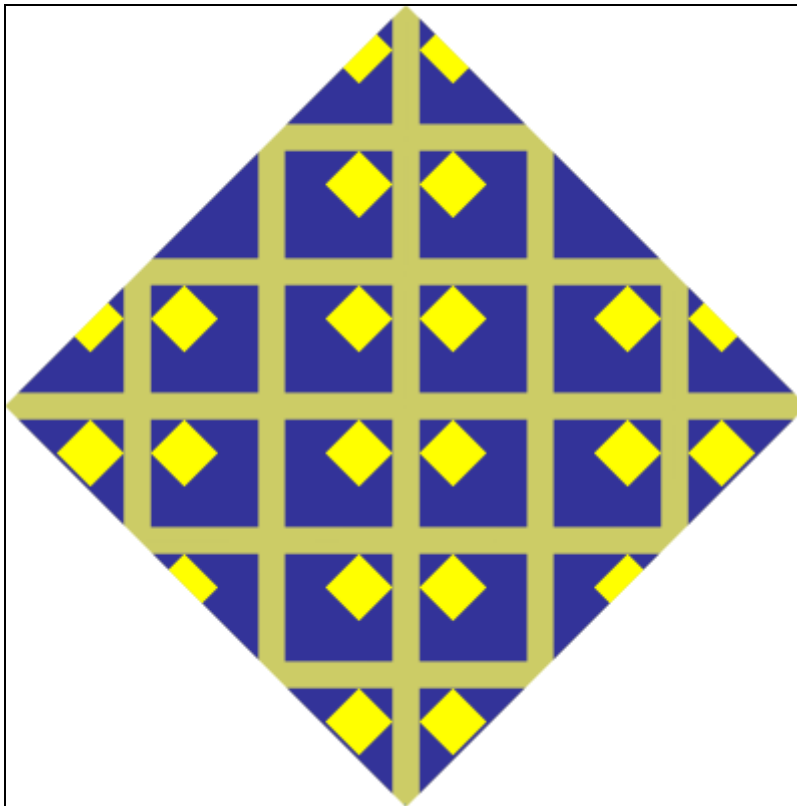
21 *end example]*

```

1  Example 13-14. Visual brush with a TileMode value of FlipX
2  <!-- Draw background diamond to show where fill affects background -->
3  <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
4  <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
5      <Path.Fill>
6          <VisualBrush
7              Viewbox="0,0,1,1"
8              Viewport="200,133,67,67"
9              ViewboxUnits="Absolute"
10             ViewportUnits="Absolute"
11             TileMode="FlipX">
12             <VisualBrush.Visual>
13                 <Canvas>
14                     <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
15                         L 0.1,0.9 Z" />
16                     <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
17                         L 0.6,0.35 L 0.35,0.6 Z" />
18                 </Canvas>
19             </VisualBrush.Visual>
20         </VisualBrush>
21     </Path.Fill>
22 </Path>

```

23 This markup is rendered as follows:



24
25 *end example]*

1 **13.4.2.4 FlipY**

2 The tile arrangement is similar to the Tile tile mode, but alternate rows of tiles are flipped
3 vertically. The base tile is positioned as specified by the viewport. Rows above and below are
4 flipped vertically.

5 *Example 13–15. Image brush with a TileMode value of FlipY*

```
6 <!-- Draw background diamond to show where fill affects background -->  
7 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />  
8 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">  
9 <Path.Fill>  
10 <ImageBrush  
11 <ImageSource="newspaper.png"  
12 <Viewbox="0,0,350,284"  
13 <Viewport="200,100,87,71"  
14 <ViewportUnits="Absolute"  
15 <ViewboxUnits="Absolute"  
16 <TileMode="FlipY" />  
17 </Path.Fill>  
18 </Path>
```

19 This markup is rendered as follows:



20

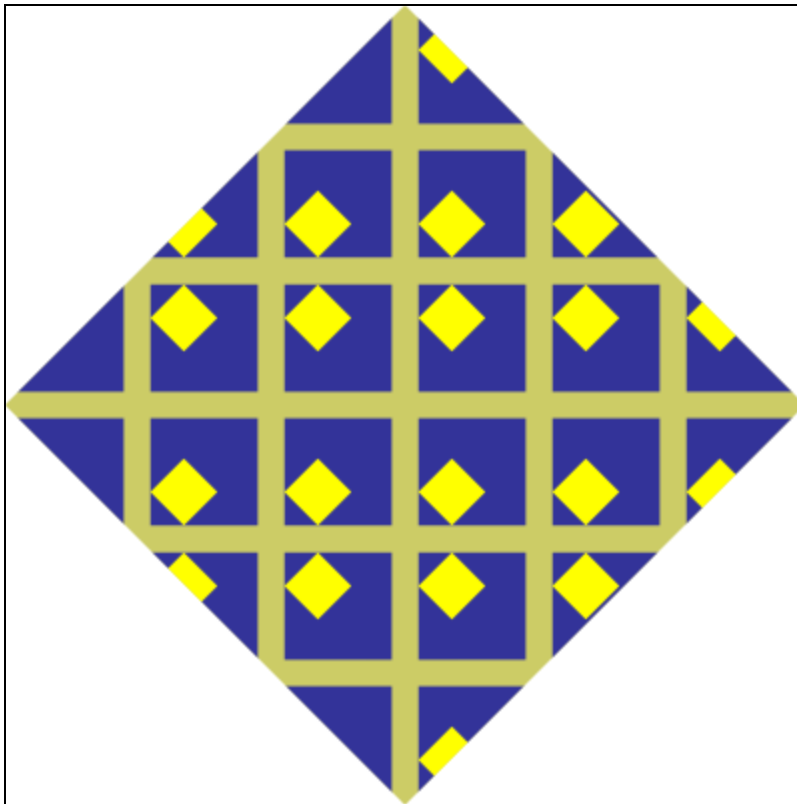
21 *end example]*

```

1  Example 13-16. Visual Brush with a TileMode value of FlipY
2  <!-- Draw background diamond to show where fill affects background -->
3  <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
4  <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
5      <Path.Fill>
6          <VisualBrush
7              Viewbox="0,0,1,1"
8              Viewport="200,133,67,67"
9              ViewboxUnits="Absolute"
10             ViewportUnits="Absolute"
11             TileMode="FlipY">
12             <VisualBrush.Visual>
13                 <Canvas>
14                     <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
15                         L 0.1,0.9 Z" />
16                     <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
17                         L 0.6,0.35 L 0.35,0.6 Z" />
18                 </Canvas>
19             </VisualBrush.Visual>
20         </VisualBrush>
21     </Path.Fill>
22 </Path>

```

23 This markup is rendered as follows:



24
25 *end example]*

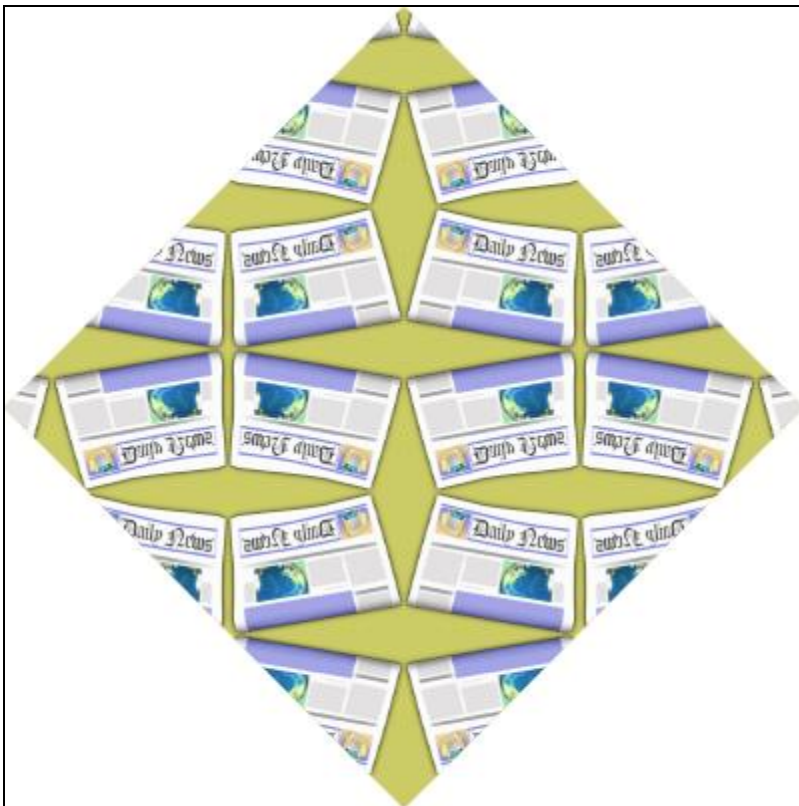
1 **13.4.2.5 FlipXY**

2 The tile arrangement is similar to the Tile tile mode, but alternate columns of tiles are flipped
 3 horizontally and alternate rows of tiles are flipped vertically. The base tile is positioned as
 4 specified by the viewport.

5 *Example 13–17. Image brush with a TileMode value of FlipXY*

```
6 <!-- Draw background diamond to show where fill affects background -->
7 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
8 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
9   <Path.Fill>
10    <ImageBrush
11      ImageSource="newspaper.png"
12      Viewbox="0,0,350,284"
13      Viewport="200,100,87,71"
14      ViewportUnits="Absolute"
15      ViewboxUnits="Absolute"
16      TileMode="FlipXY" />
17   </Path.Fill>
18 </Path>
```

19 This markup is rendered as follows:



20

21 *end example]*

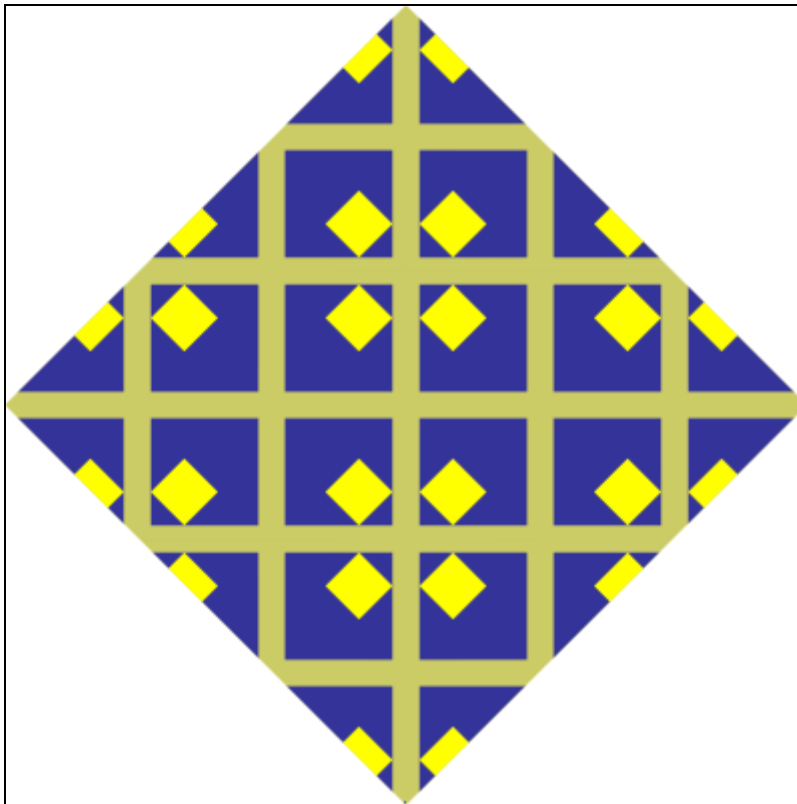
1 *Example 13-18. Visual brush with a TileMode value of FlipXY*

```

2 <!-- Draw background diamond to show where fill affects background -->
3 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
4 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
5   <Path.Fill>
6     <VisualBrush
7       Viewbox="0,0,1,1"
8       Viewport="200,133,67,67"
9       ViewboxUnits="Absolute"
10      ViewportUnits="Absolute"
11      TileMode="FlipXY">
12       <VisualBrush.Visual>
13         <Canvas>
14           <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
15             L 0.1,0.9 Z" />
16           <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
17             L 0.6,0.35 L 0.35,0.6 Z" />
18         </Canvas>
19       </VisualBrush.Visual>
20     </VisualBrush>
21   </Path.Fill>
22 </Path>

```

23 This markup is rendered as follows:

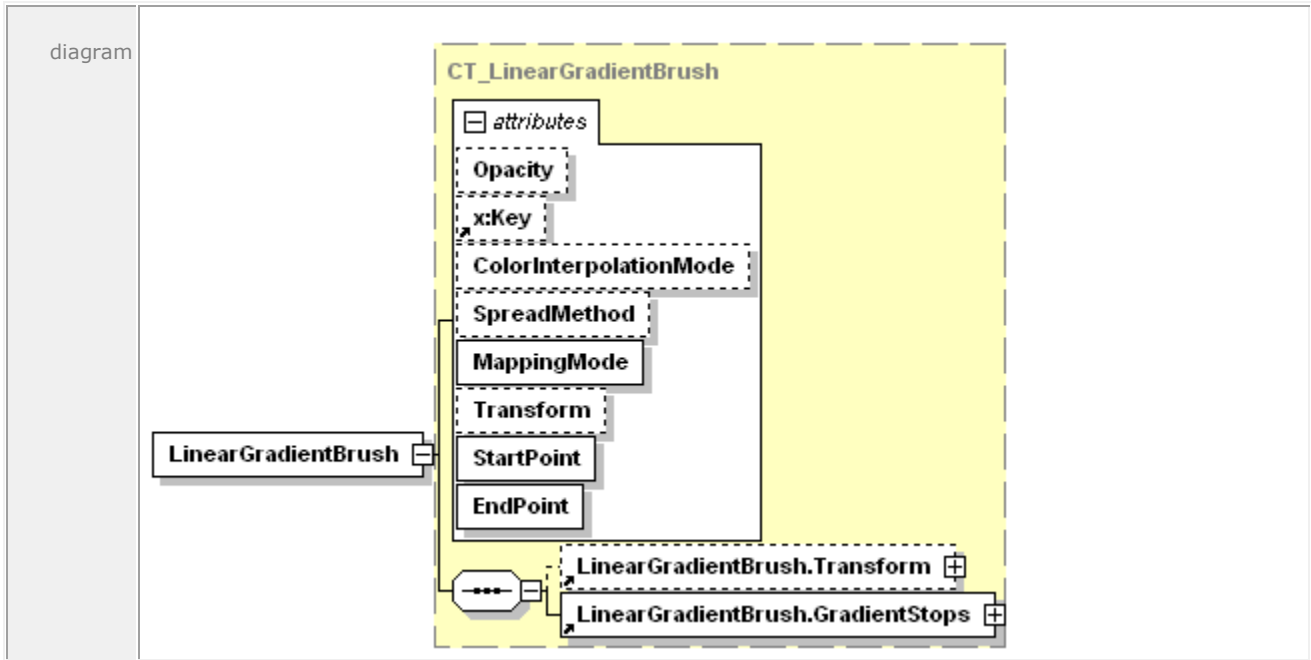


24

25 *end example]*

1 **13.5 <LinearGradientBrush> Element**

2 element **LinearGradientBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the linear gradient. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.5].
	ColorInterpolationMode	<u>ST_ClrIntMode</u>		SRgbLinear Interpolation		Specifies the gamma function for color interpolation. The gamma adjustment should not be applied to the alpha component, if specified. Valid values are SRgbLinearInterpolation and ScRgbLinearInterpolation.

SpreadMethod	ST_SpreadMethod		Pad		Describes how the brush should fill the content area outside of the primary, initial gradient area. Valid values are Pad, Reflect and Repeat.
MappingMode	ST_MappingMode	required		Absolute	Specifies that the start point and end point are defined in the effective coordinate space (includes the Transform attribute of the brush).
Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property on a brush is concatenated with the current effective render transform to yield an effective render transform local to the brush. The start point and end point are transformed using the local effective render transform.
StartPoint	ST_Point	required			Specifies the starting point of the linear gradient.
EndPoint	ST_Point	required			Specifies the end point of the linear gradient. The linear gradient brush interpolates the colors from the start point to the end point, where the start point represents an offset of 0, and the EndPoint represents an offset of 1. The Offset attribute value specified in a GradientStop element relates to the 0 and 1 offsets defined by the start point and end point.
annotation	Fills a region with a linear gradient.				

- 1 The <LinearGradientBrush> element is used to specify a linear gradient brush along a vector.
- 2 For details about computing a linear gradient, see §18.3.

3 *Example 13–19. <LinearGradientBrush> usage*

- 4 The following markup describes a page with a rectangular path that is filled with a linear gradient:

```

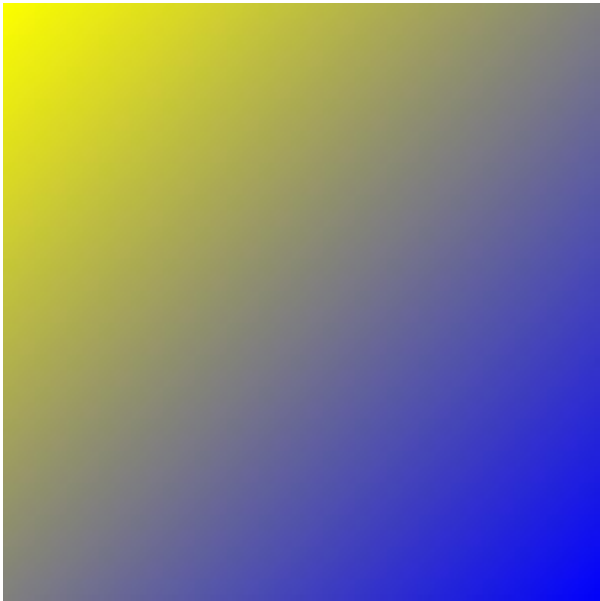
5
6     <Path>
7         <Path.Fill>
```

```

1      <LinearGradientBrush
2          MappingMode="Absolute"
3          StartPoint="0,0"
4          EndPoint="300,300">
5          <LinearGradientBrush.GradientStops>
6              <GradientStop Color="#FFFF00" Offset="0" />
7              <GradientStop Color="#0000FF" Offset="1" />
8          </LinearGradientBrush.GradientStops>
9      </LinearGradientBrush>
10     </Path.Fill>
11     <Path.Data>
12         <PathGeometry>
13             <PathFigure StartPoint="0,0">
14                 <PolyLineSegment Points="300,0 300,300 0,300" />
15             </PathFigure>
16         </PathGeometry>
17     </Path.Data>
18 </Path>

```

19 This markup is rendered as follows:



20
21 *end example]*

22 13.5.1 SpreadMethod Attribute

23 The SpreadMethod attribute describes the fill for areas beyond the start point and end point of
24 the linear gradient brush. Valid values are Pad, Reflect, and Repeat.

25 *Example 13–20. Linear gradient brush with a SpreadMethod value of Pad*

26 In this method, the first color and the last color are used to fill the remaining fill area at the
27 beginning and end.

```

28     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
29         <Path.Fill>
30             <LinearGradientBrush

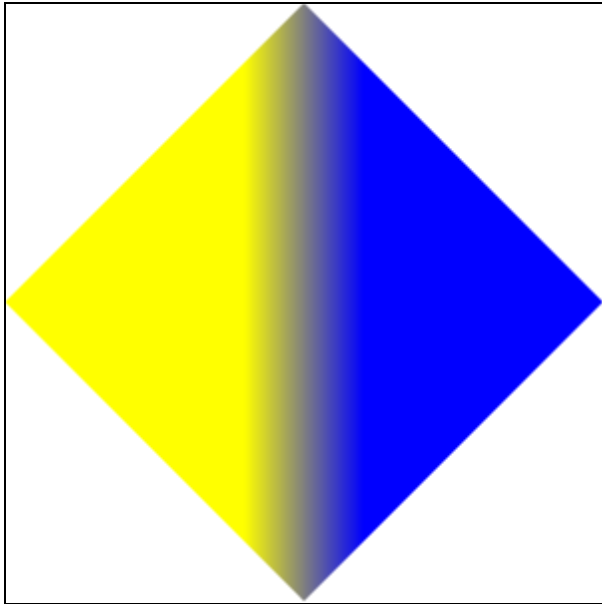
```

```

1      MappingMode="Absolute"
2      StartPoint="120,0"
3      EndPoint="180,0"
4      SpreadMethod="Pad">
5      <LinearGradientBrush.GradientStops>
6          <GradientStop Color="#FFFF00" Offset="0.0" />
7          <GradientStop Color="#0000FF" Offset="1.0" />
8      </LinearGradientBrush.GradientStops>
9      </LinearGradientBrush>
10     </Path.Fill>
11 </Path>

```

12 This markup is rendered as follows:



13

14 *end example]*

15 *Example 13–21. Linear gradient brush with a SpreadMethod value of Reflect*

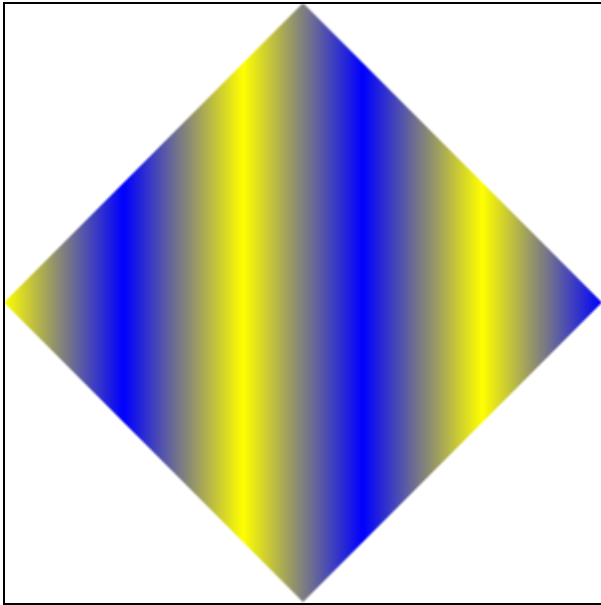
16 In this method, the gradient stops are replayed in reverse order repeatedly to cover the fill
17 area.

```

18     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
19         <Path.Fill>
20             <LinearGradientBrush
21                 MappingMode="Absolute"
22                 StartPoint="120,0"
23                 EndPoint="180,0"
24                 SpreadMethod="Reflect">
25                 <LinearGradientBrush.GradientStops>
26                     <GradientStop Color="#FFFF00" Offset="0.0" />
27                     <GradientStop Color="#0000FF" Offset="1.0" />
28                 </LinearGradientBrush.GradientStops>
29             </LinearGradientBrush>
30         </Path.Fill>
31 </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13-22. Linear gradient brush with a SpreadMethod value of Repeat*

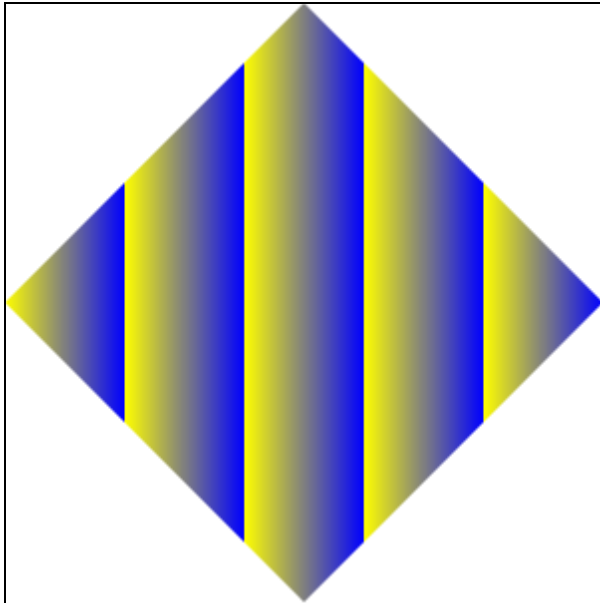
5 In this method, the gradient stops are repeated in order until the fill area is covered.

```

6     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
7         <Path.Fill>
8             <LinearGradientBrush
9                 MappingMode="Absolute"
10                StartPoint="120,0"
11                EndPoint="180,0"
12                SpreadMethod="Repeat">
13                <LinearGradientBrush.GradientStops>
14                    <GradientStop Color="#FFFF00" Offset="0.0" />
15                    <GradientStop Color="#0000FF" Offset="1.0" />
16                </LinearGradientBrush.GradientStops>
17            </LinearGradientBrush>
18        </Path.Fill>
19    </Path>

```

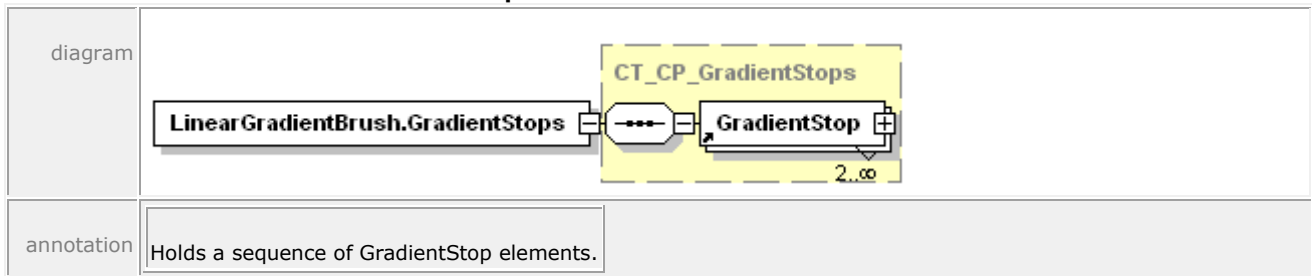
1 This markup is rendered as follows:



2
3 *end example]*

4 **13.5.2 <LinearGradientBrush.GradientStops> Element**

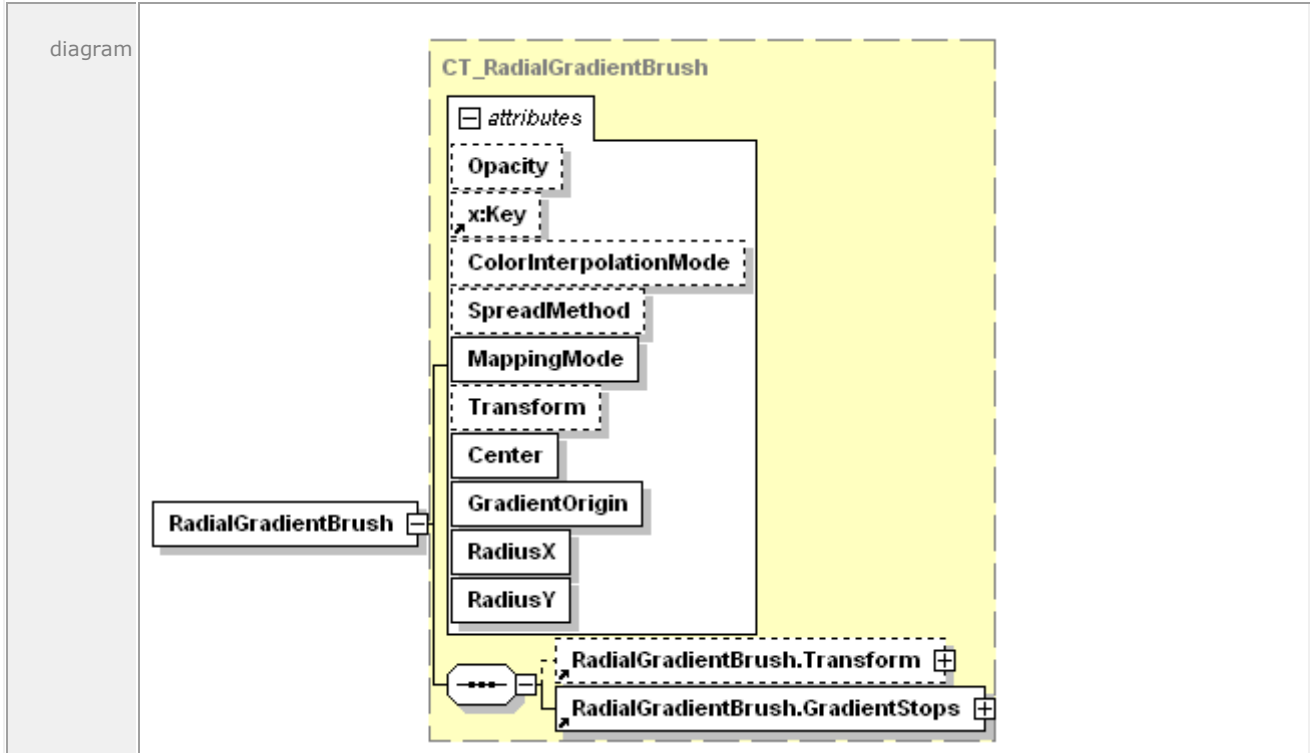
5 element **LinearGradientBrush.GradientStops**



6
7 The <LinearGradientBrush.GradientStops> property element specifies a collection of gradient
8 stops that comprise the linear gradient. For more information, see §13.7.

1 **13.6 <RadialGradientBrush> Element**

2 element **RadialGradientBrush**



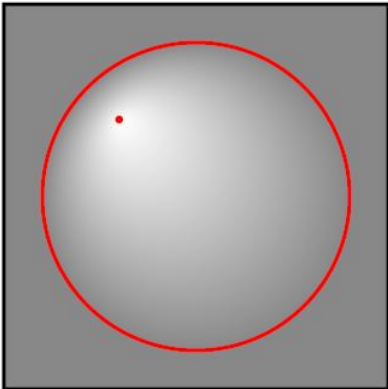
attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the radial gradient. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.6].
	ColorInterpolationMode	<u>ST_ClrIntMode</u>		SRGBLinear Interpolation		Specifies the gamma function for color interpolation for sRGB colors. The gamma adjustment should not be applied to the alpha component, if specified. Valid values are

					SRgbLinearInterpolation and ScRgbLinearInterpolation.
SpreadMethod	ST_SpreadMethod		Pad		Describes how the brush should fill the content area outside of the primary, initial gradient area. Valid values are Pad, Reflect and Repeat.
MappingMode	ST_MappingMode	required		Absolute	Specifies that center, x radius, and y radius are defined in the effective coordinate space (includes the Transform attribute of the brush).
Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The ellipse defined by the center, gradient origin, x radius, and y radius values is transformed using the local effective render transform.
Center	ST_Point	required			Specifies the center point of the radial gradient (that is, the center of the ellipse). The radial gradient brush interpolates the colors from the gradient origin to the circumference of the ellipse. The circumference is determined by the center and the radii.
GradientOrigin	ST_Point	required			Specifies the origin point of the radial gradient.
RadiusX	ST_GEZero	required			Specifies the radius in the x dimension of the ellipse which defines the radial gradient.
RadiusY	ST_GEZero	required			Specifies the radius in the y dimension of the ellipse which defines the radial gradient.
annotation	Fills a region with a radial gradient.				

- 1 Radial gradient brushes are similar to linear gradient brushes. However, whereas a linear
- 2 gradient brush has a start point and end point to define the gradient vector, a radial gradient
- 3 brush has an ellipse (defined by the center, x radius, and y radius) and a gradient origin. The
- 4 ellipse defines the end point of the gradient. In other words, a gradient stop with an offset
- 5 at 1.0 defines the color at the circumference of the ellipse. The gradient origin defines the
- 6 center of the gradient. A gradient stop with an offset at 0.0 defines the color at the gradient
- 7 origin.
- 8 For details about computing a radial gradient, see §18.3.3.

1 *Example 13–23. A radial gradient brush*

2 The following figure is a radial gradient that transitions from white to gray. The outside ellipse
 3 represents the gradient ellipse while the dot denotes the gradient origin. This gradient has a
 4 SpreadMethod value of Pad. *end example]*



5

6 *end example]*

7 *Example 13–24. RadialGradientBrush usage*

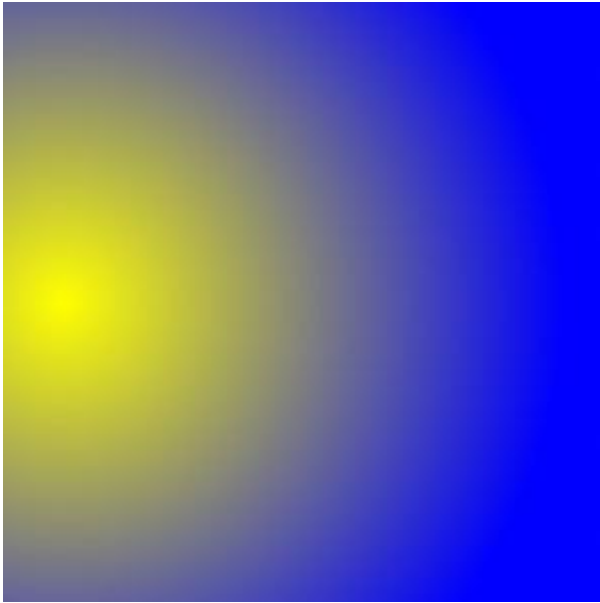
8 The following markup describes a page with a rectangular path that is filled with a radial
 9 gradient:

```

10 <Path>
11   <Path.Fill>
12     <RadialGradientBrush
13       MappingMode="Absolute"
14       Center="30,150"
15       GradientOrigin="30,150"
16       RadiusX="250"
17       RadiusY="250">
18       <RadialGradientBrush.GradientStops>
19         <GradientStop Color="#FFFF00" Offset="0" />
20         <GradientStop Color="#0000FF" Offset="1" />
21       </RadialGradientBrush.GradientStops>
22     </RadialGradientBrush>
23   </Path.Fill>
24   <Path.Data>
25     <PathGeometry>
26       <PathFigure StartPoint="0,0" IsClosed="true">
27         <PolyLineSegment Points="300,0 300,300 0,300" />
28       </PathFigure>
29     </PathGeometry>
30   </Path.Data>
31 </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 **13.6.1 SpreadMethod Attribute**

5 The SpreadMethod attribute describes the fill of areas beyond the ellipse described by the center,
6 x radius, and y radius of the radial gradient brush. Valid values are Pad, Reflect, and Repeat.

7 *Example 13–25. Radial gradient brush with a SpreadMethod value of Pad*

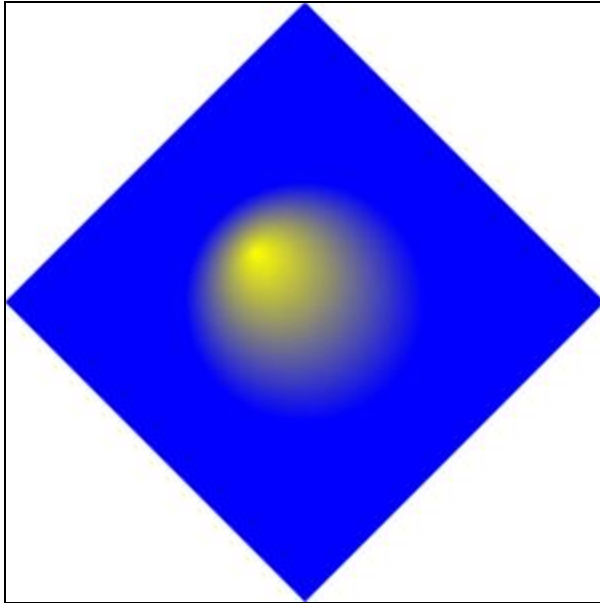
8 In the following markup, the last color is used to cover the fill area outside the ellipse.

```

9     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
10         <Path.Fill>
11             <RadialGradientBrush
12                 MappingMode="Absolute"
13                 Center="150,150"
14                 GradientOrigin="125,125"
15                 RadiusX="60"
16                 RadiusY="60"
17                 SpreadMethod="Pad">
18                 <RadialGradientBrush.GradientStops>
19                     <GradientStop Color="#FFFF00" Offset="0.0" />
20                     <GradientStop Color="#0000FF" Offset="1.0" />
21                 </RadialGradientBrush.GradientStops>
22             </RadialGradientBrush>
23         </Path.Fill>
24     </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13–26. Radial gradient brush with a SpreadMethod value of Reflect*

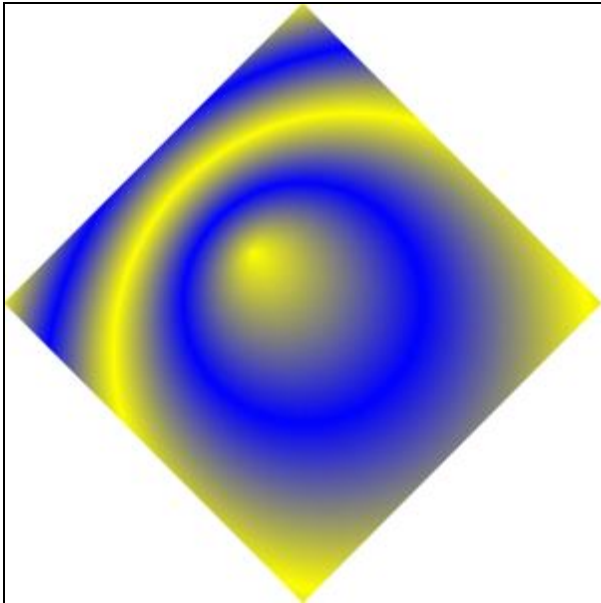
5 In the following markup, the gradient stops are replayed in reverse order repeatedly to cover
6 the fill area.

```

7     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
8         <Path.Fill>
9             <RadialGradientBrush
10                MappingMode="Absolute"
11                Center="150,150"
12                GradientOrigin="125,125"
13                RadiusX="60"
14                RadiusY="60"
15                SpreadMethod="Reflect">
16                <RadialGradientBrush.GradientStops>
17                    <GradientStop Color="#FFFF00" Offset="0.0" />
18                    <GradientStop Color="#0000FF" Offset="1.0" />
19                </RadialGradientBrush.GradientStops>
20            </RadialGradientBrush>
21        </Path.Fill>
22    </Path>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 *Example 13-27. Radial gradient brush with a SpreadMethod value of Repeat*

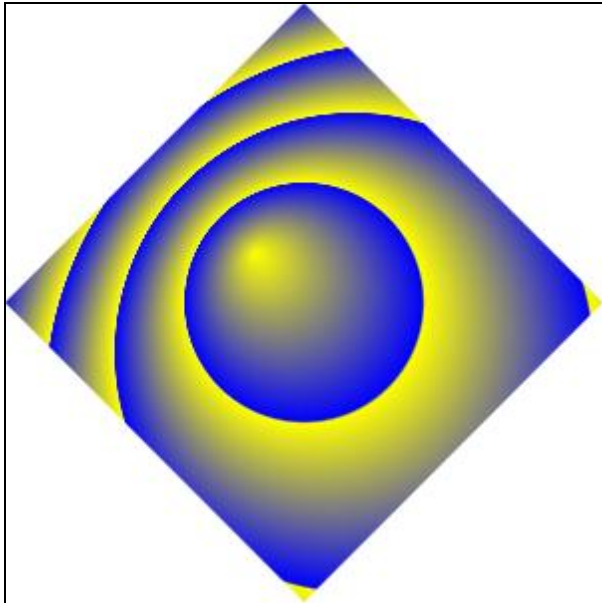
5 In the following markup, the gradient stops are repeated in order until the fill area is covered.

```

6     <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
7         <Path.Fill>
8             <RadialGradientBrush
9                 MappingMode="Absolute"
10                Center="150,150"
11                GradientOrigin="125,125"
12                RadiusX="60"
13                RadiusY="60"
14                SpreadMethod="Repeat">
15                <RadialGradientBrush.GradientStops>
16                    <GradientStop Color="#FFFF00" Offset="0.0" />
17                    <GradientStop Color="#0000FF" Offset="1.0" />
18                </RadialGradientBrush.GradientStops>
19            </RadialGradientBrush>
20        </Path.Fill>
21    </Path>

```

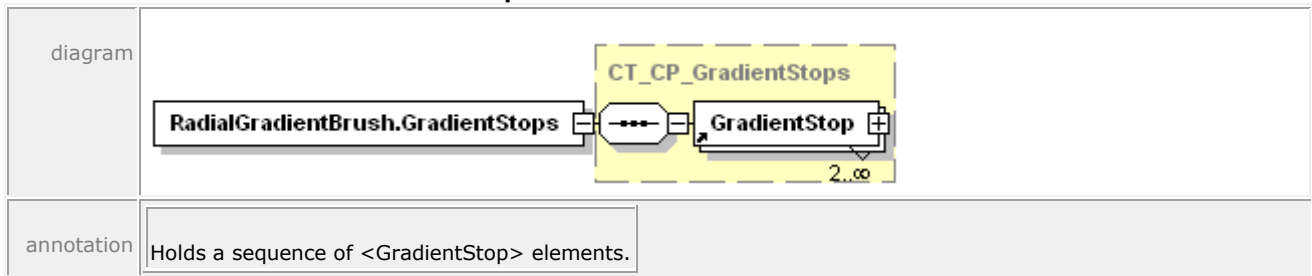
1 This markup is rendered as follows:



2
3 *end example]*

4 **13.6.2 <RadialGradientBrush.GradientStops> Element**

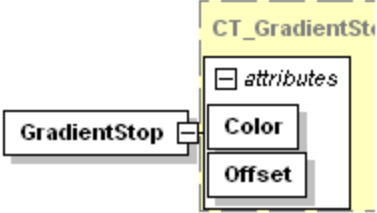
5 element **RadialGradientBrush.GradientStops**



6 The <RadialGradientBrush.GradientStops> property element specifies a collection of gradient
7 stops that comprise the radial gradient. For more information, see §13.7.

1 13.7 <GradientStop> Element

2 element **GradientStop**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Color	<u>ST_Color</u>	required			Specifies the gradient stop color. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.
	Offset	<u>ST_Double</u>	required			Specifies the gradient offset. The offset indicates a point along the progression of the gradient at which a color is specified. Colors between gradient offsets in the progression are interpolated.
annotation	Indicates a location and range of color progression for rendering a gradient.					

3 The <GradientStop> element is used by both the <LinearGradientBrush> and
 4 <RadialGradientBrush> elements to define the location and range of color progression for
 5 rendering a gradient.

6 For linear gradient brushes, the offset value of 0.0 is mapped to the start point of the gradient,
 7 and the offset value of 1.0 is mapped to the end point. Intermediate offset values are
 8 interpolated between these two points to determine their location.

9 For radial gradient brushes, the offset value of 0.0 is mapped to the gradient origin location.
 10 The offset value of 1.0 is mapped to the circumference of the ellipse as determined by the
 11 center, x radius, and y radius. Offsets between 0.0 and 1.0 are positioned at a location
 12 interpolated between these points.

13 For full details of rendering of gradient brushes, including handling of offsets, please see §18.3.

1 13.8 Using a Brush as an Opacity Mask

2 Each pixel carries an alpha value ranging from 0.0 (fully transparent) to 1.0 (fully opaque). The
3 alpha value is used when blending elements to achieve the visual effect of transparency. Each
4 element can have an Opacity attribute by which the alpha value of each pixel is multiplied
5 uniformly.

6 The OpacityMask property also allows the specification of per-pixel opacity, which controls how
7 rendered content is blended with its destination. The opacity specified by the opacity mask is
8 combined multiplicatively with any opacity that can already be present in the alpha channel of
9 the contents. The per-pixel opacity specified by the opacity mask is determined by the alpha
10 channel of each pixel in the mask. The color data is ignored.

11 The alpha value of the area not marked by the brush is 0.0. The required computations for
12 transparently blending two elements when rendering, also known as *alpha blending*, are
13 described in §18.4.

14 An opacity mask always has a brush child element.

15 *Example 13–28. Opacity mask with linear gradient*

16 The following markup illustrates how an opacity mask is used to create a fade effect on a glyph.
17 The opacity mask is a linear gradient that fades from opaque black to transparent black.

```
18 <FixedPage Height="1056" Width="816" xml:lang="en-US">
19   <Glyphs
20     OriginX="25"
21     OriginY="50"
22     UnicodeString="This is a fading text example."
23     FontUri=" ../Resources/Fonts/Times.TTF"
24     FontRenderingEmSize="32">
25     <Glyphs.OpacityMask>
26       <LinearGradientBrush
27         StartPoint="25,0"
28         EndPoint="450,0"
29         MappingMode="Absolute">
30         <LinearGradientBrush.GradientStops>
31           <GradientStop Color="#FF000000" Offset="0" />
32           <GradientStop Color="#00000000" Offset="1" />
33         </LinearGradientBrush.GradientStops>
34       </LinearGradientBrush>
35     </Glyphs.OpacityMask>
36     <Glyphs.Fill>
37       <SolidColorBrush Color="#000000" />
38     </Glyphs.Fill>
39   </Glyphs>
40 </FixedPage>
```

1 This markup is rendered as follows:

2 *This is a fading text example.*

3 *end example]*

1 *Example 13–29. Opacity mask with radial gradient*

2 In the following markup, the opacity mask is a radial gradient:

```
3     <FixedPage Width="816" Height="1056" xml:lang="en-US">
4         <Path>
5             <Path.OpacityMask>
6                 <RadialGradientBrush
7                     MappingMode="Absolute"
8                     Center="200,300"
9                     GradientOrigin="200,300"
10                    RadiusX="200"
11                    RadiusY="300">
12                    <RadialGradientBrush.GradientStops>
13                        <GradientStop Color="#FF000000" Offset="0" />
14                        <GradientStop Color="#20000000" Offset="1" />
15                    </RadialGradientBrush.GradientStops>
16                </RadialGradientBrush>
17            </Path.OpacityMask>
18            <Path.Fill>
19                <ImageBrush
20                    Viewbox="0,0,400,600"
21                    ViewboxUnits="Absolute"
22                    Viewport="0,0,400,600"
23                    ViewportUnits="Absolute"
24                    TileMode="None"
25                    ImageSource="images/jpeg3.jpg" />
26            </Path.Fill>
27            <Path.Data>
28                <PathGeometry>
29                    <PathFigure StartPoint="0,0" IsClosed="true">
30                        <PolyLineSegment Points="400,0 400,600 0,600" />
31                    </PathFigure>
32                </PathGeometry>
33            </Path.Data>
34        </Path>
35    </FixedPage>
```

1 This markup is rendered as follows:



2

3 *end example]*

1 14. Common Properties

2 Several XPS Document elements share property attributes and elements as summarized in
 3 Table 14–1 and Table 14–2 and detailed in the following sections. Other than the Name,
 4 FixedPage.NavigateUri, and xml:lang attributes, these properties compose their results from
 5 parent to child, as described in §18.4.1.

6 *Table 14–1. Common property attributes*

Name	Applies to	Description
Clip	<Canvas> <Glyphs> <Path>	Restricts the region to which a brush can be applied.
Opacity	<Canvas> <Glyphs> <ImageBrush> <LinearGradientBrush> <Path> <RadialGradientBrush> <SolidColorBrush> <VisualBrush>	Defines the uniform transparency of the element.
OpacityMask	<Canvas> <Glyphs> <Path>	Specifies a mask of alpha values.
RenderTransform	<Canvas> <Glyphs> <Path>	Establishes a new coordinate space through the use of an affine matrix transformation. For more information, see §14.4.
Transform	<ImageBrush> <LinearGradientBrush> <PathGeometry> <RadialGradientBrush> <VisualBrush>	Establishes a new coordinate space through the use of an affine matrix transformation. Geometry transformations are applied before brushes. The results are concatenated with any containing effective render transformation specification.
Name	<Canvas> <FixedPage> <Glyphs> <Path>	Defines a hyperlink target or identifies an element uniquely for document structure markup to reference. For more information, see §16.2.
FixedPage.NavigateUri	<Canvas> <Glyphs> <Path>	Defines a hyperlink source. For more information, see §16.2.

xml:lang	<Canvas> <FixedPage> <Glyphs> <Path>	Specifies a language.
----------	---	-----------------------

1 Table 14–2. Common property elements

Name	Description
<Canvas.Resources> <FixedPage.Resources>	Contains elements that can be reused by reference throughout the markup of the <FixedPage> or <Canvas> child or descendant elements.
<Canvas.Clip> <Glyphs.Clip> <Path.Clip>	Restricts the region to which a brush can be applied.
<Canvas.RenderTransform> <Glyphs.RenderTransform> <Path.RenderTransform>	Establishes a new coordinate space through the use of an affine matrix transformation. For more information, see §14.4.
<ImageBrush.Transform> <LinearGradientBrush.Transform> <PathGeometry.Transform> <RadialGradientBrush.Transform> <VisualBrush.Transform>	Establishes a new effective coordinate space through the use of an affine matrix transformation. Path geometry transformations (<PathGeometry.Transform>) are applied before brushes. The results are concatenated with any containing effective render transformation.
<Canvas.OpacityMask> <Glyphs.OpacityMask> <Path.OpacityMask>	Specifies a mask of alpha values that is applied in the same fashion as the Opacity attribute, but allows different alpha values on a pixel-by-pixel basis.

2 **14.1 Opacity**

3 The Opacity property attribute is used to transparently blend the current element with
4 previously specified elements, also known as alpha blending. The opacity value MUST fall within
5 the 0 (fully transparent) to 1 (fully opaque) range, inclusive [M2.72].

6 For more information, see §18.4.

7 **14.2 Resources and Resource References**

8 Fixed page markup supports the concept of resources. A *resource* is a reusable property value
9 that is expressed in markup, identified by a *key*, and stored in a *resource dictionary*. In general,
10 any property value that can be expressed using property element syntax can be held in a
11 resource dictionary.

12 Each resource in a resource dictionary has a key. Any property that specifies its value by
13 referencing a resource key in a resource dictionary is called a *resource reference*.

1 The <Canvas> and <FixedPage> elements can carry a resource dictionary. A resource
 2 dictionary is expressed in markup by the <FixedPage.Resources> or <Canvas.Resources>
 3 property element. Individual resource values MUST be specified within a resource dictionary
 4 [M7.1].

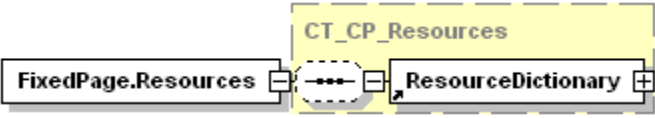
5 The <Canvas.Resources> or <FixedPage.Resources> property elements MUST precede any
 6 property elements of the <Canvas> or <FixedPage> elements [M2.72]. Likewise, they MUST
 7 precede any path, glyphs, or canvas children of the <Canvas> or <FixedPage> elements
 8 [M2.72].

9 Alternatively, resource dictionaries MAY be specified in separate parts and referenced from
 10 within the <FixedPage.Resources> or <Canvas.Resources> property element [O7.1]. Such a
 11 *remote resource dictionary* can be shared across multiple pages. [*Example: By defining a brush*
 12 *in a remote resource dictionary, graphical elements that are common to multiple pages can be*
 13 *reused. end example*]

14 The <Path>, <Glyphs>, and <Canvas> elements can appear as a resource definition solely for
 15 the purpose of using these elements in the Visual attribute of a <VisualBrush> element.
 16 Brushes and geometries appear in resource dictionaries far more frequently.

17 14.2.1 <FixedPage.Resources> Element

18 element **FixedPage.Resources**

diagram	 <p>The diagram shows a box labeled 'FixedPage.Resources' containing a box labeled 'ResourceDictionary'. A dashed box labeled 'CT_CP_Resources' encloses the 'ResourceDictionary' box. A dashed arrow points from the 'FixedPage.Resources' box to the 'ResourceDictionary' box.</p>
annotation	Contains the resource dictionary for the <FixedPage> element.

19 *Example 14-1. <FixedPage.Resources> usage*

```

20 <FixedPage Width="816" Height="1056" xml:lang="en-US"
21   xmlns="http://schemas.microsoft.com/xps/2005/06"
22   xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
23     key">
24   <FixedPage.Resources>
25     <ResourceDictionary>
26       <PathGeometry x:Key="Rectangle">
27         <PathFigure StartPoint="20,20" IsClosed="true">
28           <PolyLineSegment Points="120,20 120,70 20,70" />
29         </PathFigure>
30       </PathGeometry>
31     </ResourceDictionary>
32   </FixedPage.Resources>
33   <Path Stroke="#000000"
34     StrokeThickness="1"
35     Data="{StaticResource Rectangle}" />
36 </FixedPage>

```

1 This markup is rendered as follows:

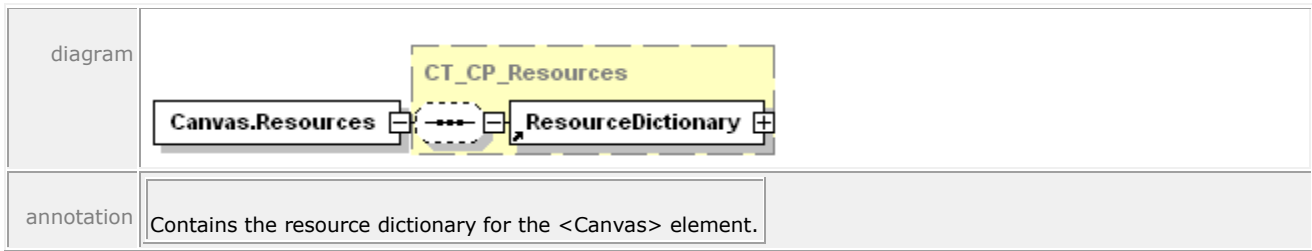


2

3 *end example]*

1 14.2.2 <Canvas.Resources> Element

2 element **Canvas.Resources**



3

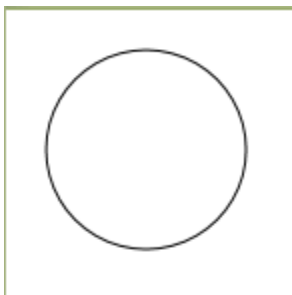
4 *Example 14–2. <Canvas.Resources> usage*

```

5     <Canvas
6         xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
7             key">
8         <Canvas.Resources>
9             <ResourceDictionary>
10                <PathGeometry x:Key="Circle">
11                    <PathFigure StartPoint="20,70">
12                        <ArcSegment
13                            Point="120,70"
14                            Size="50,50"
15                            RotationAngle="0"
16                            IsLargeArc="true"
17                            SweepDirection="Clockwise" />
18                        <ArcSegment
19                            Point="20,70"
20                            Size="50,50"
21                            RotationAngle="0"
22                            IsLargeArc="true"
23                            SweepDirection="Clockwise" />
24                    </PathFigure>
25                </PathGeometry>
26            </ResourceDictionary>
27        </Canvas.Resources>
28        <Path Stroke="#000000"
29            StrokeThickness="1"
30            Data="{StaticResource Circle}" />
31    </Canvas>

```

32 This markup is rendered as follows:

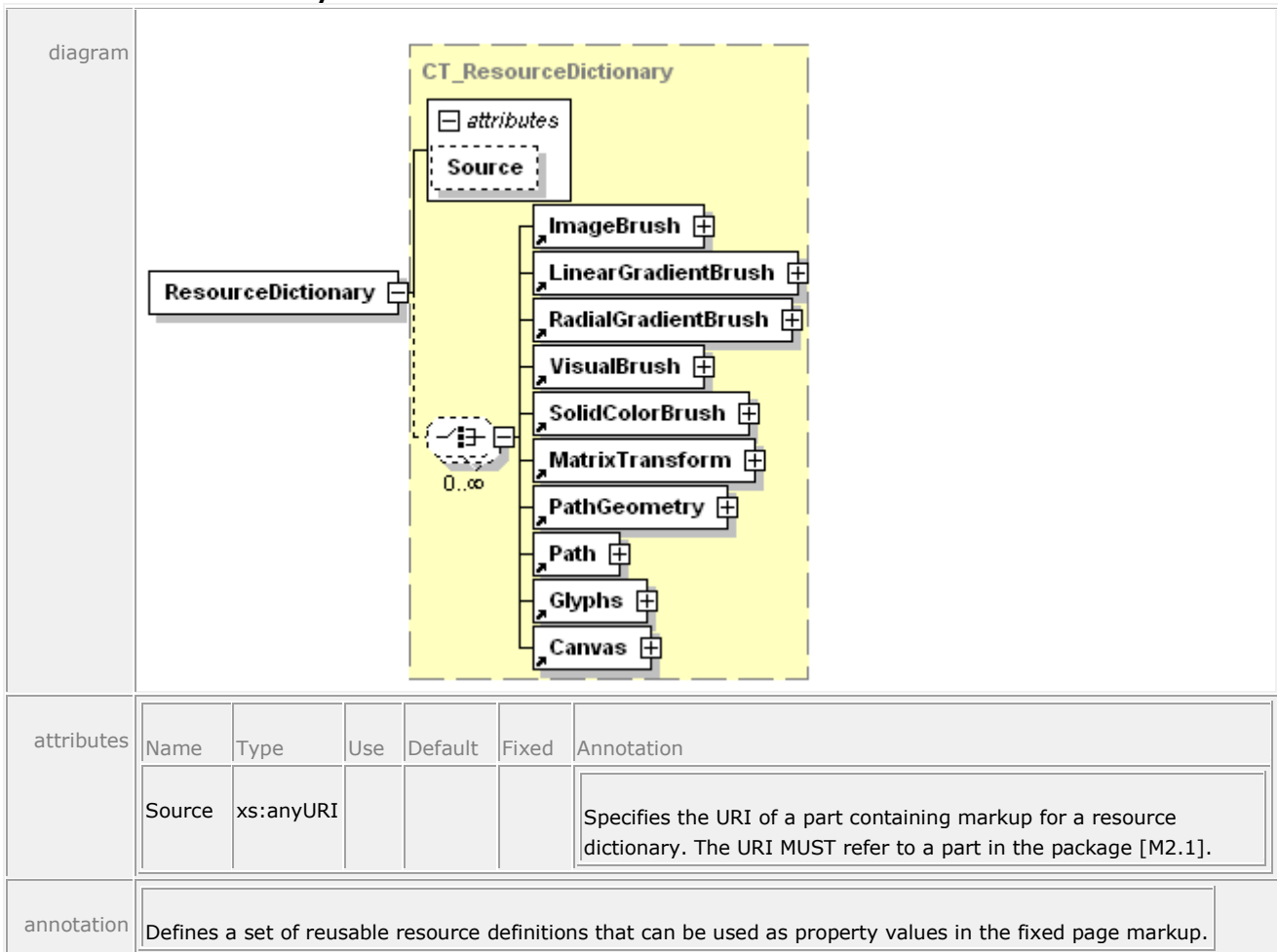


33

34 *end example]*

1 **14.2.3 <ResourceDictionary> Element**

2 element **ResourceDictionary**



3 The <FixedPage.Resources> and <Canvas.Resources> property elements contain exactly one
 4 <ResourceDictionary> element. A resource dictionary contains *resource definition* element
 5 entries. Each resource definition has a key specified in the x:Key attribute that is unique within
 6 the scope of the resource dictionary. The x:Key attribute is included in the Resource Dictionary
 7 namespace specified in §H.

8 Resource dictionaries can be declared inline inside a <FixedPage.Resources> or
 9 <Canvas.Resources> element, or they MAY be defined in a separate part and referenced by a
 10 <ResourceDictionary> element inside a <FixedPage.Resources> or <Canvas.Resources>
 11 element [O7.1]. This allows resource dictionaries to be shared across parts. [Example: A single
 12 resource dictionary can be used by every fixed page in the XPS Document. end example]
 13 See §14.2.3.1 for more details.

14 A resource definition MAY reference another resource defined previously in the same resource
 15 dictionary [O7.2]. If the resource dictionary does not appear in a separate part, a resource
 16 definition MAY reference a previously defined resource in a resource dictionary of a parent or
 17 ancestor <Canvas> or <FixedPage> element [O7.3].

1 Namespace prefixes in resource definitions MUST apply in the context of the definition, rather
 2 than in the context of the resource reference [M7.2]. An `xml:lang` attribute within a resource
 3 definition MUST be interpreted in the context of the resource reference, not the resource
 4 definition [M7.3].

5 *Example 14-3. Resource dictionary markup*

6 The following markup defines two geometries, one for a rectangle, and the other for a circle:

```

7     <Canvas
8         xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
9             key">
10        <Canvas.Resources>
11            <ResourceDictionary>
12                <PathGeometry x:Key="Rectangle">
13                    <PathFigure StartPoint="20,20" IsClosed="true">
14                        <PolyLineSegment Points="120,20 120,70 20,70" />
15                    </PathFigure>
16                </PathGeometry>
17                <PathGeometry x:Key="Circle">
18                    <PathFigure StartPoint="20,70">
19                        <ArcSegment
20                            Point="120,70"
21                            Size="50,50"
22                            RotationAngle="0"
23                            IsLargeArc="true"
24                            SweepDirection="Clockwise" />
25                        <ArcSegment
26                            Point="20,70"
27                            Size="50,50"
28                            RotationAngle="0"
29                            IsLargeArc="true"
30                            SweepDirection="Clockwise" />
31                    </PathFigure>
32                </PathGeometry>
33            </ResourceDictionary>
34        </Canvas.Resources>
35        <Path Data="{StaticResource Rectangle}">
36            <Path.Fill>
37                <SolidColorBrush Color="#FF0000" />
38            </Path.Fill>
39        </Path>
40    </Canvas>

```

41 *end example]*

42 **14.2.3.1 Remote Resource Dictionaries**

43 A resource dictionary MAY be defined in a separate part [O7.1]. This is referred to as a *remote*
 44 *resource dictionary*. A remote resource dictionary MUST follow the requirements above that
 45 apply to all resource dictionaries [M7.4]. A remote resource dictionary MUST NOT contain any
 46 resource definition children that reference another remote resource dictionary [M7.5].

47 The `<FixedPage.Resources>` and `<Canvas.Resources>` property elements include a remote
 48 resource dictionary via reference, using the `Source` attribute of the `<ResourceDictionary>`
 49 element.

1 A <ResourceDictionary> element that specifies a remote resource dictionary in its Source
 2 attribute MUST NOT contain any resource definition children [M7.6]. <FixedPage.Resources>
 3 and <Canvas.Resources> elements that include a remote resource dictionary MUST include
 4 exactly one <ResourceDictionary> element [M2.72].

5 A remote Resource Dictionary part MUST be added as a Required Resource relationship from
 6 the FixedPage part that references it [M2.10]. In addition, producers MUST add each resource
 7 such as fonts or images referenced in the Resource Dictionary part as a Required Resource
 8 relationship from the FixedPage part (*not* the Resource Dictionary part) to the indirectly
 9 required resource, even if the particular fixed page does not reference the resource [M2.10].
 10 For more information, see §H.3.

11 Inline references to fonts or images in remote resource dictionary entries MUST be interpreted
 12 with the same base URI as the Remote Resource Dictionary part, not from the base URI of the
 13 part referring to the particular remote resource dictionary entry [M7.7].

14 *Example 14–4. A remote resource dictionary and reference*

15 The following markup defines a resource dictionary that contains two geometries, one for a
 16 rectangle and the other for a circle:

```

17 <!-- Contents of /resource.xaml -->
18 <ResourceDictionary xmlns="http://schemas.microsoft.com/xps/2005/06"
19   xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
20   key">
21   <PathGeometry x:Key="Rectangle">
22     <PathFigure StartPoint="20,20" IsClosed="true">
23       <PolyLineSegment Points="120,20 120,70 20,70" />
24     </PathFigure>
25   </PathGeometry>
26   <PathGeometry x:Key="Circle">
27     <PathFigure StartPoint="20,70">
28       <ArcSegment
29         Point="120,70"
30         Size="50,50"
31         RotationAngle="0"
32         IsLargeArc="true"
33         SweepDirection="Clockwise" />
34       <ArcSegment
35         Point="20,70"
36         Size="50,50"
37         RotationAngle="0"
38         IsLargeArc="true"
39         SweepDirection="Clockwise" />
40     </PathFigure>
41   </PathGeometry>
42 </ResourceDictionary>

```

43 The following markup references the previously defined resource dictionary:

```

44 <Canvas
45   xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
46   key">
47   <Canvas.Resources>
48     <ResourceDictionary Source="/resource.xaml"/>
49   </Canvas.Resources>

```

```
1      <Path Data="{StaticResource Rectangle}">
2          <Path.Fill>
3              <SolidColorBrush Color="#FF0000" />
4          </Path.Fill>
5      </Path>
6  </Canvas>
7  end example]
```

1 **14.2.4 Resource References**

2 To set a property value to a defined resource, use the form:

3 {StaticResource *key*}

4 Where *key* is the same string specified with x:Key in the resource definition.

5 The context of the resource reference determines how defined resources are rendered (such as
6 the transformation matrix to be applied). Specifically, the effective coordinate space for
7 rendering the referenced resource is a composition of the effective coordinate space of the
8 referring element plus any Transform or RenderTransform properties included in the resource
9 definition itself.

10 It is considered an error if a static resource reference cannot be resolved, or if it *can* be
11 resolved but the resource type does not match the usage at the location of reference.

12 *Example 14–5. Using a resource reference to fill a brush*

13 In the following markup, the rectangular region defined by the geometry specified in the
14 dictionary is filled by a solid color brush:

```

15     <Canvas
16         xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
17             key">
18         <Canvas.Resources>
19             <ResourceDictionary>
20                 <PathGeometry x:Key="Rectangle">
21                     <PathFigure StartPoint="20,20" IsClosed="true">
22                         <PolyLineSegment Points="120,20 120,70 20,70" />
23                     </PathFigure>
24                 </PathGeometry>
25             </ResourceDictionary>
26         </Canvas.Resources>
27         <Path Data="{StaticResource Rectangle}">
28             <Path.Fill>
29                 <SolidColorBrush Color="#FF0000" />
30             </Path.Fill>
31         </Path>
32     </Canvas>

```

33 *end example]*

34 **14.2.5 Scoping Rules for Resolving Resource References**

35 The value of the x:Key attribute **MUST** be unique within the resource dictionary [M2.72].
36 However, the resource dictionary of a <Canvas> element **MAY** re-use an x:Key value defined in
37 the resource dictionary of a parent or ancestor <Canvas> or <FixedPage> element [O7.5].
38 Resource references are resolved from the innermost to the outermost resource dictionary.

39 A resource definition **MAY** reference a previously defined resource with the same name that is
40 defined in an ancestor resource dictionary [O7.6]; the reference **MUST** be resolved before the
41 redefined resource is added to the dictionary [M7.8].

42 A resource definition **MAY** reference another resource defined prior to the point of reference,
43 including a resource previously defined within the same resource dictionary [O7.2]. If a

1 resource definition references another resource, the reference MUST be resolved in the context
2 of the resource definition, not in the context of the resource use [M7.9].

3 To find a resource, the nearest parent or ancestor canvas or fixed page is searched. If the
4 desired name is not defined in the initially searched resource dictionary, then the next-nearest
5 parent or ancestor canvas or fixed page is searched. An error occurs if the search has continued
6 to the root <FixedPage> element and a specified resource has not been found. This search
7 occurs only within the containing FixedPage part.

8 *Example 14–6. Using scoping rules*

```

9     <FixedPage
10         xmlns="http://schemas.microsoft.com/xps/2005/06"
11         xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
12             key"
13         Height="1056" Width="816" xml:lang="en-US">
14         <FixedPage.Resources>
15             <ResourceDictionary>
16                 <SolidColorBrush x:Key="FavoriteColorFill" Color="#808080" />
17             </ResourceDictionary>
18         </FixedPage.Resources>
19         <Canvas>
20             <Canvas.Resources>
21                 <ResourceDictionary>
22                     <SolidColorBrush x:Key="FavoriteColorFill"
23                         Color="#000000" />
24                 </ResourceDictionary>
25             </Canvas.Resources>
26             <!-- The following path is filed with color #000000 -->
27             <Path Fill="{StaticResource FavoriteColorFill}">
28                 <Path.Data>
29                     ...
30                 </Path.Data>
31             </Path>
32         </Canvas>
33         <!-- The following path is filed with color #000000 -->
34         <Path Fill="{StaticResource FavoriteColorFill}">
35             <Path.Data>
36                 ...
37             </Path.Data>
38         </Path>
39     </Canvas>
40 </FixedPage>
41 <!-- The following path is filled with color #808080 -->
42 <Path Fill="{StaticResource FavoriteColorFill}">
43     <Path.Data>
44         ...
45     </Path.Data>
46 </Path>
47 </FixedPage>
48 end example]

```

1 14.2.6 Support for Markup Compatibility

2 If a resource dictionary contains Markup Compatibility and Extensibility elements and
3 attributes, the processing of the Markup Compatibility and Extensibility markup **MUST** occur in
4 the context of the definition of the resource dictionary, not in the context of resource references
5 [M2.10].

14.3 Clipping

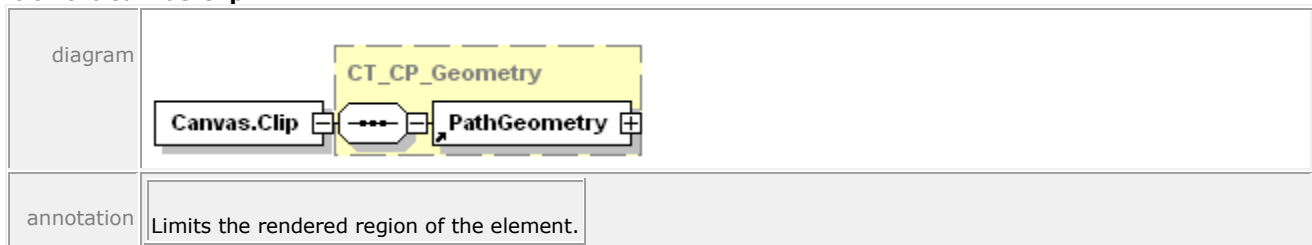
The Clip property specifies a geometric area that restricts the rendered region of an element.

The geometry is specified by a child `<PathGeometry>` element as detailed in §11.2, or by abbreviated geometry syntax, described in §11.2.3.

The default fill rule for geometries that do not specify a value is EvenOdd.

14.3.1 `<Canvas.Clip>` Element

element **Canvas.Clip**



The `<Canvas.Clip>` property element applies to all child and descendant elements of the canvas.

Example 14–7. Canvas clip markup and rendering

```

11     <Canvas>
12         <Canvas.Clip>
13             <PathGeometry>
14                 <PathFigure StartPoint="25,25" IsClosed="true">
15                     <PolyLineSegment Points="60,25 70,60 80,25 115,25
16                         115,115 80,115 70,80 60,115 25,115" />
17                 </PathFigure>
18             </PathGeometry>
19         </Canvas.Clip>
20         <Path Fill="#9999CC">
21             <Path.Data>
22                 <PathGeometry>
23                     <PathFigure StartPoint="20,70">
24                         <ArcSegment
25                             Point="120,70"
26                             Size="50,50"
27                             RotationAngle="0"
28                             IsLargeArc="true"
29                             SweepDirection="Clockwise" />
30                     <ArcSegment
31                         Point="20,70"
32                         Size="50,50"
33                         RotationAngle="0"
34                         IsLargeArc="true"
35                         SweepDirection="Clockwise" />
36                     </PathFigure>
37                 </PathGeometry>
38             </Path.Data>

```

```

1     </Path>
2     </Canvas>

```

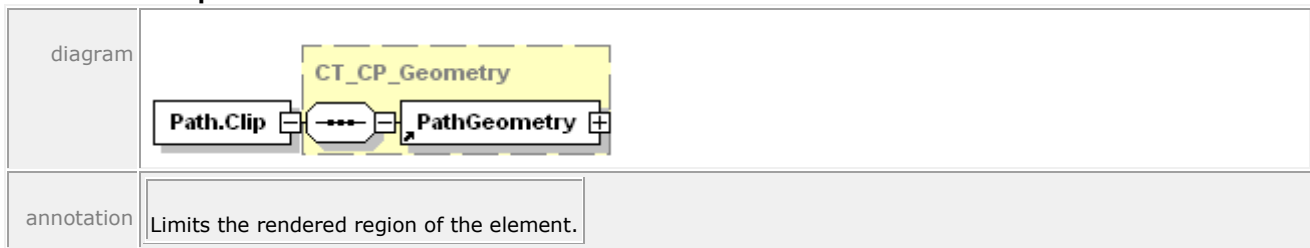
3 This markup is rendered as follows:



4
5 *end example]*

6 14.3.2 <Path.Clip> Element

7 element **Path.Clip**



8 A clipping region can also be applied to a specific path.

9 *Example 14-8. <Path.Clip> usage*

10 The following markup describes a complex clipping behavior:

```

11     <Path Fill="#9999CC">
12         <Path.Clip>
13             <PathGeometry>
14                 <PathFigure StartPoint="25,25" IsClosed="true">
15                     <PolyLineSegment Points="115,25 115,115 25,115" />
16                 </PathFigure>
17                 <PathFigure StartPoint="55,55" IsClosed="true">
18                     <PolyLineSegment Points="85,55 85,85 55,85" />
19                 </PathFigure>
20             </PathGeometry>
21         </Path.Clip>
22         <Path.Data>
23             <PathGeometry>
24                 <PathFigure StartPoint="20,70">
25                     <ArcSegment
26                         Point="120,70"
27                         Size="50,50"
28                         RotationAngle="0"
29                         IsLargeArc="true"
30                         SweepDirection="Clockwise" />
31                     <ArcSegment
32                         Point="20,70"
33                         Size="50,50"

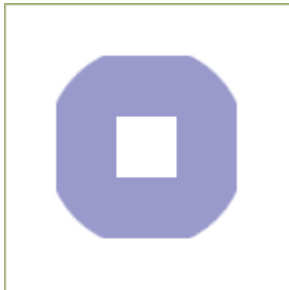
```

```

1           RotationAngle="0"
2           IsLargeArc="true"
3           SweepDirection="Clockwise" />
4       </PathFigure>
5   </PathGeometry>
6 </Path.Data>
7 </Path>

```

8 This markup is rendered as follows:



9

10 *end example]*

11 14.3.3 <Glyphs.Clip> Element

12 element **Glyphs.Clip**

diagram	
annotation	<p>Limits the rendered region of the element. Only portions of the <Glyphs> element that fall within the clip region (even partially clipped characters) produce marks on the page.</p>

13 *Example 14–9. <Glyphs.Clip> usage*

14 The following markup uses abbreviated geometry syntax to define the clipping region:

```

15 <Glyphs
16     Fill="#000000"
17     Clip="M 0,0 L 180,0 L 180,140 L 0,140 Z M 20,60 L 140,60 L 140,80
18         L 20,80 Z"
19     OriginX="20"
20     OriginY="130"
21     UnicodeString="N"
22     FontRenderingEmSize="170"
23     FontUri="../Resources/Fonts/Timesbd.ttf" />

```

1 This markup is rendered as follows:



2

3 *end example]*

4 **14.4 Positioning Content**

5 Content is positioned according to the properties specified for the fixed page or canvas, the
 6 properties specified for elements within the fixed page or canvas, and the compositional rules
 7 defined for the fixed payload namespace.

8 Elements are positioned relative to the current origin (0,0) of the coordinate space. The current
 9 origin can be moved by setting the RenderTransform property of a canvas, path, or glyph. The
 10 render transformation establishes a new coordinate frame for all children of the parent element.

11 Geometries and brushes can be manipulated in a similar way by setting the Transform
 12 property. The transform results are concatenated with the current render transformation to
 13 create an effective render transformation for the local element.

14 The RenderTransform and Transform properties both specify an affine matrix transformation to
 15 the local coordinate space, using the <MatrixTransform> element as their value. An
 16 abbreviated matrix transformation syntax MAY be used to specify a RenderTransform or Transform
 17 attribute value [M2.72].

18 **14.4.1 <MatrixTransform> Element**

19 element **MatrixTransform**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Matrix	ST_Matrix	required			Specifies the matrix structure that defines the transformation.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a

					resource dictionary [M7.11].
annotation	Creates an arbitrary affine matrix transformation that manipulates objects or coordinate systems in a two-dimensional plane.				

1 The <MatrixTransform> element defines an arbitrary affine matrix transformation used to
 2 manipulate the coordinate systems of elements. A 3x3 matrix is used for transformations in an
 3 x,y plane. Affine transformation matrices can be multiplied to form any number of linear
 4 transformations, such as rotation and skew (shear), followed by translation. An affine
 5 transformation matrix has its final column equal to 0,0,1, so only the members in the first two
 6 columns are specified.

$$7 \begin{bmatrix} M11 & M12 & 0 \\ M21 & M22 & 0 \\ \text{OffsetX} & \text{OffsetY} & 1 \end{bmatrix}$$

8 This structure is specified by the Matrix attribute of the <MatrixTransform> element as the six
 9 numbers in the first two columns. [*Example*: "M11,M12,M21,M22,OffsetX,OffsetY". *end*
 10 *example*]

11 A matrix transform can also be specified as a RenderTransform or Transform property attribute
 12 using the following abbreviated matrix transformation syntax:

13 M11,M12,M21,M22,OffsetX,OffsetY

14 The values M11, M12, M21, and M22 control linear transformations such as rotation and skew,
 15 while OffsetX and OffsetY provide positional translation. Some typical affine matrix
 16 transformation examples follow.

17 *Example 14–10. Matrix scaling*

$$18 \begin{bmatrix} \text{X scale-factor} & 0 & 0 \\ 0 & \text{Y scale-factor} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

19 *end example*]

20 *Example 14–11. Matrix reversing the x axis*

$$21 \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

22 *end example*]

1 *Example 14-12. Matrix reversing the y axis*

$$2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3 *end example]*

4 *Example 14-13. Matrix skewing*

$$5 \begin{bmatrix} 1 & \text{Y skew-factor} & 0 \\ \text{X skew factor} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

6 *end example]*

7 *Example 14-14. Matrix Rotating*

$$8 \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

9 *end example]*

10 *Example 14-15. Matrix positioning*

$$11 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \text{OffsetX} & \text{OffsetY} & 1 \end{bmatrix}$$

12 *end example]*

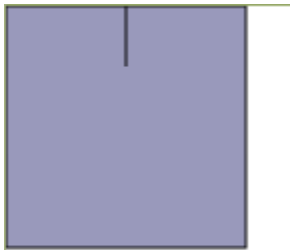
13 *Example 14-16. <MatrixTransform> usage*

14 The following markup describes a box (with the top edge marked) that is rotated 90° and
15 shifted 50 units down and to the right:

```
16 <Path
17   Stroke="#000000"
18   Fill="#9999BB"
19   Data="M 0,0 L 60,0 L 60,25 L 60,0 L 120,0 L 120,120 L 0,120 Z">
20   <Path.RenderTransform>
21     <MatrixTransform Matrix="0,1,-1,0,170,50" />
22   </Path.RenderTransform>
23 </Path>
```

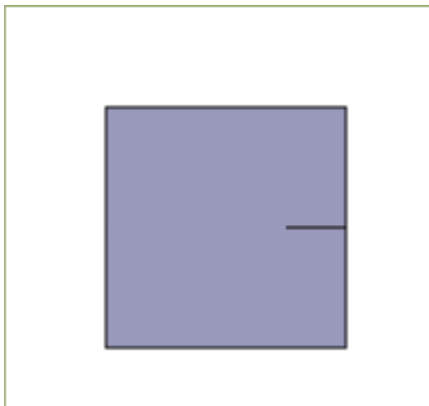
24 Since the x origin has been shifted, the overall box must be additionally shifted the width of the
25 box to achieve the desired visual effect.

1 Before the render transformation, the box appears like this:



2

3 After the render transformation, the box appears like this:



4

5 *end example]*

1 *Example 14–17. Using abbreviated matrix transformation syntax*

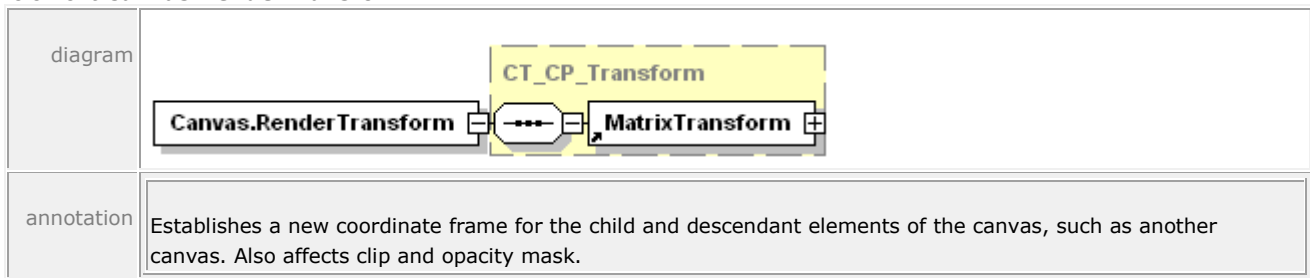
2 The following markup uses abbreviated syntax to produce the above image:

```
3 <Path
4   Stroke="#000000"
5   Fill="#9999BB"
6   Data="M 0,0 L 60,0 L 60,25 L 60,0 L 120,0 L 120,120 L 0,120 Z"
7   RenderTransform="0,1, -1,0,170,50" />
```

8 *end example]*

9 **14.4.2 <Canvas.RenderTransform> Element**

10 element **Canvas.RenderTransform**



11 *Example 14–18. <Canvas.RenderTransform> usage*

12 In the following markup, child elements of the canvas are positioned by the render
13 transformation:

```
14 <Canvas>
15   <Canvas.Resources>
16     <ResourceDictionary>
17       <PathGeometry x:Key="StarFish">
18         <PathFigure StartPoint="50,0" IsClosed="true">
19           <PolyLineSegment Points="55,45 100,25 55,50 80,100 50,55
20             20,100 45,50 0,25 45,45" />
21         </PathFigure>
22       </PathGeometry>
23     </ResourceDictionary>
24   </Canvas.Resources>
25
26   <!-- Draw a green starfish shifted 25 to the right and 50 down -->
27   <Canvas>
28     <Canvas.RenderTransform>
29       <MatrixTransform Matrix="1,0,0,1,25,50" />
30     </Canvas.RenderTransform>
31     <Path Data="{StaticResource StarFish}">
32       <Path.Fill>
33         <SolidColorBrush Color="#00FF00" />
34       </Path.Fill>
35     </Path>
36   </Canvas>
37
38   <!-- Draw a red starfish shifted 100 to the right and 150 down -->
39   <Canvas>
```



```

1      <Canvas.RenderTransform>
2          <MatrixTransform Matrix="1,0,0,1,100,150" />
3      </Canvas.RenderTransform>
4      <Path Data="{StaticResource StarFish}">
5          <Path.Fill>
6              <SolidColorBrush Color="#FF0000" />
7          </Path.Fill>
8      </Path>
9  </Canvas>
10 </Canvas>

```

11 This markup is rendered as follows:

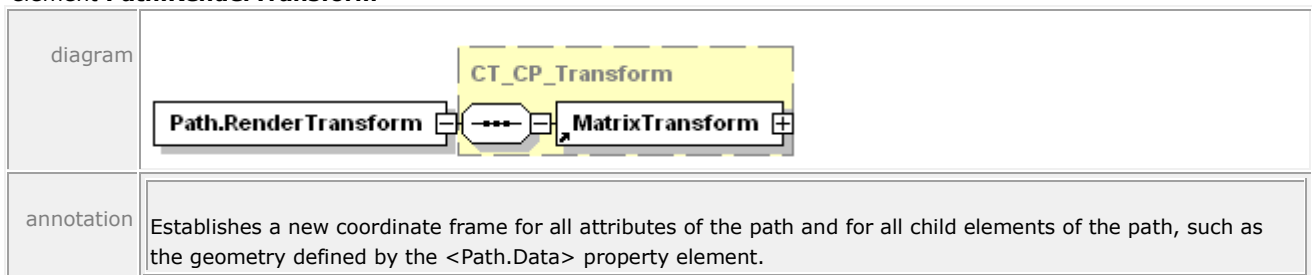


12

13 *end example]*

14 14.4.3 <Path.RenderTransform> Element

15 element **Path.RenderTransform**



16 *Example 14-19. <Path.RenderTransform> usage*

17 The following markup describes a y-skew transformation applied to a circular path. (Before the
 18 render transformation, the middle of the right edge of the circle was marked with a horizontal
 19 line.)

```

20 <Path
21     Fill="#999999"
22     Stroke="#000000"
23     Data="M 20,70 A 50,50 0 1 1 120,70 L 100,70 L 120,70 A 50,50 0 1 1
24         20,70 Z" >

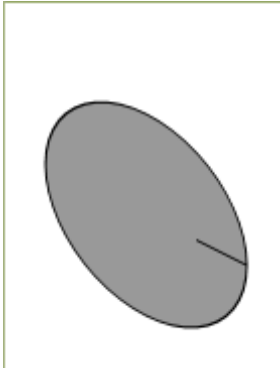
```

```

1      <Path.RenderTransform>
2          <MatrixTransform Matrix="1,0.5,0,1,0,0" />
3      </Path.RenderTransform>
4  </Path>

```

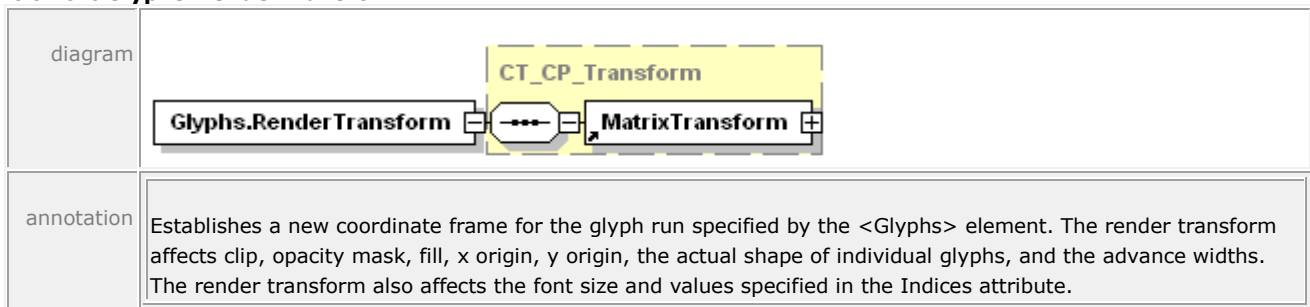
5 This markup is rendered as follows:



6
7 *end example]*

8 **14.4.4 <Glyphs.RenderTransform> Element**

9 element **Glyphs.RenderTransform**



10 *Example 14-20. <Glyphs.RenderTransform> usage*

11 The following markup describes the letter J, flipped vertically and repositioned.

```

12  <Glyphs
13      Fill="#000000"
14      OriginX="20"
15      OriginY="130"
16      UnicodeString="J"
17      FontRenderingEmSize="170"
18      FontUri="../Resources/Fonts/Timesbd.ttf" >
19      <Glyphs.RenderTransform>
20          <MatrixTransform Matrix="1,0,0,-1,0,150" />
21      </Glyphs.RenderTransform>
22  </Glyphs>

```

1 This markup is rendered as follows:



2

3 *end example]*

4 **14.4.5 <PathGeometry.Transform> Element**

5 element **PathGeometry.Transform**

diagram	
annotation	<p>Specifies the local matrix transformation that is applied to all child and descendant elements of the path geometry before it is used for filling, clipping, or stroking.</p>

6 *Example 14–21. <PathGeometry.Transform> usage*

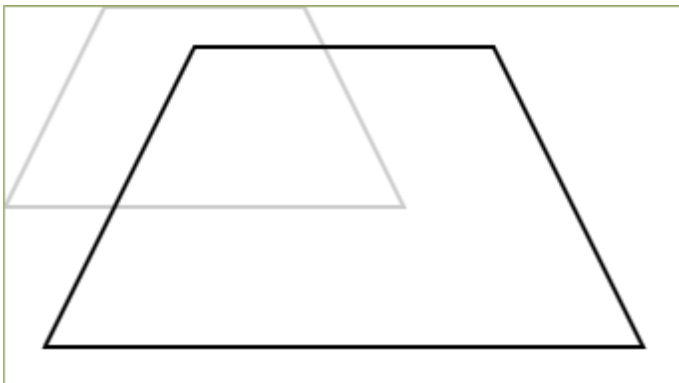
7 The following markup demonstrates a simple 150% zoom and positional transformation:

```

8   <Path StrokeThickness="2" Stroke="#000000">
9     <Path.Data>
10      <PathGeometry>
11        <PathGeometry.Transform>
12          <MatrixTransform Matrix="1.5,0,0,1.5,20,20" />
13        </PathGeometry.Transform>
14        <PathFigure StartPoint="50,0" IsClosed="true">
15          <PolyLineSegment Points="150,0 200,100 0,100" />
16        </PathFigure>
17      </PathGeometry>
18    </Path.Data>
19  </Path>

```

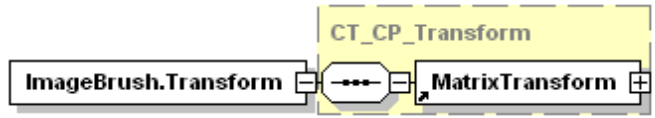
- 1 This markup is rendered as follows. The pre-transform path is indicated in light gray. Note that
- 2 the stroke thickness did not change. If this transformation had been applied to the entire Path,
- 3 the stroke thickness would also have increased by 150%.



- 4
- 5 *end example]*

1 **14.4.6 <ImageBrush.Transform> Element**

2 element **ImageBrush.Transform**

diagram	
annotation	<p>Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.</p>

3 The Transform property can result in a non-rectangular (that is, skewed) viewport that defines
 4 the tile shape. In this circumstance, tile mode operations (FlipX, FlipY, and FlipXY) are treated
 5 as if the tile was rectangular, a larger tile was constructed from a 2-by-2 arrangement of
 6 regular tiles, the skew transform was applied afterward, and the new non-rectangular tile was
 7 tiled with adjacent edges and without flipping.

8 *Example 14-22. <ImageBrush.Transform> usage*

9 The following markup describes an image rotated 20° and repositioned within a path. The path
 10 itself remains untransformed; the viewport of the image brush is transformed instead.

```

11 <Path
12   StrokeThickness="5"
13   Stroke="#996666"
14   StrokeLineJoin="Round"
15   Data="M 25,25 L 350,25 L 355,250 L 25,250 Z">
16   <Path.Fill>
17     <ImageBrush
18       ImageSource="dog.jpg"
19       TileMode="Tile"
20       Viewbox="0,0,270,423"
21       ViewboxUnits="Absolute"
22       Viewport="75,75,90,125"
23       ViewportUnits="Absolute" >
24     <ImageBrush.Transform>
25       <MatrixTransform Matrix=".939,.342,-.342,.939,0,-80" />
26     </ImageBrush.Transform>
27   </ImageBrush>
28 </Path.Fill>
29 </Path>

```

1 This markup is rendered as follows:



2
3 *end example]*

4 **14.4.7 <VisualBrush.Transform> Element**

5 element **VisualBrush.Transform**

diagram	
annotation	<p>Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.</p>

6 The Transform property can result in a non-rectangular (that is, skewed) viewport that defines
7 the tile shape. In this circumstance, tile mode operations (FlipX, FlipY, and FlipXY) are treated
8 as if the tile was rectangular, a larger tile was constructed from a 2-by-2 arrangement of
9 regular tiles, the skew transform was applied afterward, and the new non-rectangular tile was
10 tiled with adjacent edges and without flipping.

11 *Example 14-23. <VisualBrush.Transform> usage*

12 The following markup describes a solid background and vertical pinstripe rotated 45° to fill a
13 frame:

```

14 <Path
15   StrokeThickness="5"
16   Stroke="#336666"
17   StrokeLineJoin="Round"
18   Data="M 25,25 L 365,25 L 365,250 L 25,250 Z M 70,70 L 320,70
19     L 320,205 L 70,205 Z">
20 <Path.Fill>
21   <VisualBrush
22     TileMode="Tile"

```

```
1      Viewbox="0,0,60,100"  
2      ViewboxUnits="Absolute"  
3      Viewport="25,25,50,50"  
4      ViewportUnits="Absolute">  
5      <VisualBrush.Transform>  
6          <MatrixTransform Matrix=".707,.707,-.707,.707,0,0" />  
7      </VisualBrush.Transform>  
8      <VisualBrush.Visual>  
9          <Canvas>  
10             <Path  
11                 Fill="#99CCCC"  
12                 Data="M 0,0 L 60,0 L 60,100 L 0,100 Z" />  
13             <Path  
14                 Stroke="#336666"  
15                 Data="M 0,0 L 0,100 M 20,0 L 20,100 M 40,0 L 40,100  
16                     M 60,0 L 60,100 M 80,0 L 80,100" />  
17             </Canvas>  
18         </VisualBrush.Visual>  
19     </VisualBrush>  
20 </Path.Fill>  
21 </Path>
```

22 This markup is rendered as follows:



23

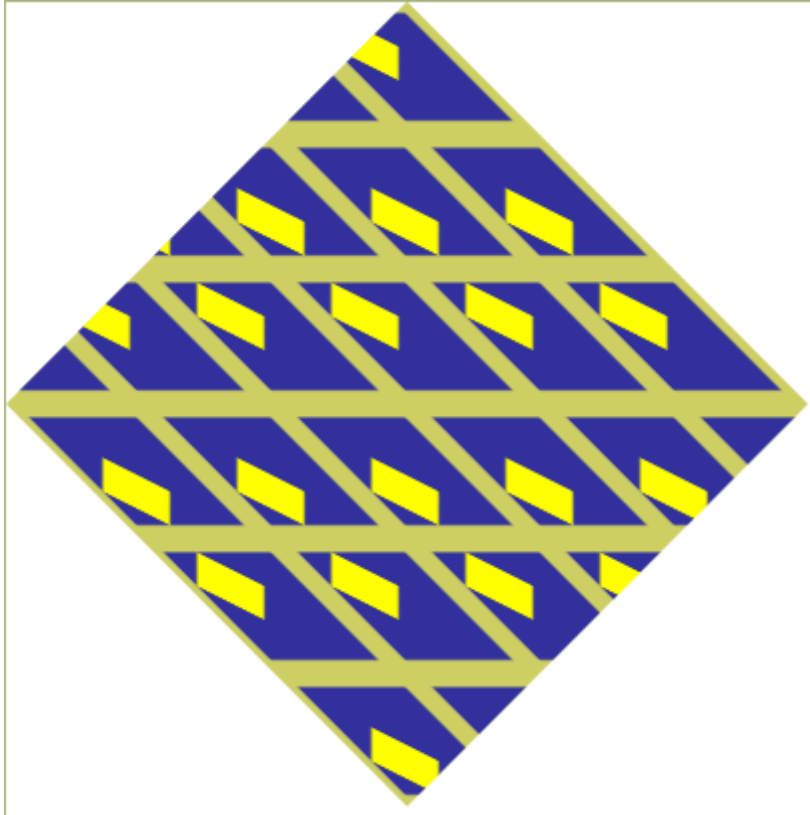
24 *end example]*

1 *Example 14-24. <VisualBrush.Transform> usage with tiling behavior*

2 This example demonstrates tile rendering behavior when applying a transform.

```
3 <!-- Draw background diamond to show where fill affects background -->
4 <Path Fill="#CCCC66" Data="M 200,0 L 400,200 L 200,400 L 0,200 Z" />
5 <Path Data="M 200,0 L 400,200 L 200,400 L 0,200 Z">
6   <Path.Fill>
7     <VisualBrush
8       Viewbox="0,0,1,1"
9       Viewport="200,133,67,67"
10      ViewboxUnits="Absolute"
11      ViewportUnits="Absolute"
12      TileMode="FlipY">
13     <VisualBrush.Transform>
14       <MatrixTransform Matrix="1,0,1,1,0,0" />
15     </VisualBrush.Transform>
16     <VisualBrush.Visual>
17       <Canvas>
18         <Path Fill="#333399" Data="M 0.1,0.1 L 0.9,0.1 L 0.9,0.9
19           L 0.1,0.9 Z" />
20         <Path Fill="#FFFF00" Data="M 0.1,0.35 L 0.35,0.1
21           L 0.6,0.35 L 0.35,0.6 Z" />
22       </Canvas>
23     </VisualBrush.Visual>
24   </VisualBrush>
25 </Path.Fill>
26 </Path>
```

27 This markup is rendered as follows:



1

2 *end example]*3 **14.4.8 <LinearGradientBrush.Transform> Element**4 element **LinearGradientBrush.Transform**

diagram	
annotation	<p>Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The start point and end point are transformed using the local effective render transform.</p>

5 *Example 14–25. <LinearGradientBrush.Transform> usage*

6 The following markup demonstrates a transform applied to the brush directly:

```

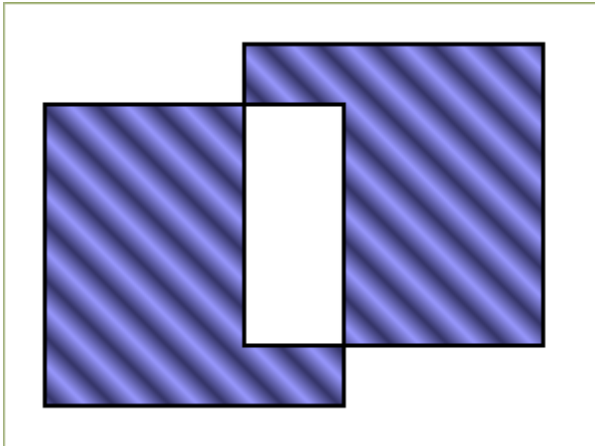
7     <Path Stroke="#000000" StrokeThickness="2" Data="M 20,50 L 170,50 L 170,200 L
8     20,200 Z M 120,20 L 270,20 L 270,170 120,170 Z">
9         <Path.Fill>
10            <LinearGradientBrush
11                MappingMode="Absolute"
12                StartPoint="0,0"
13                EndPoint="0,10"
14                SpreadMethod="Reflect">
15                <LinearGradientBrush.Transform>

```

```

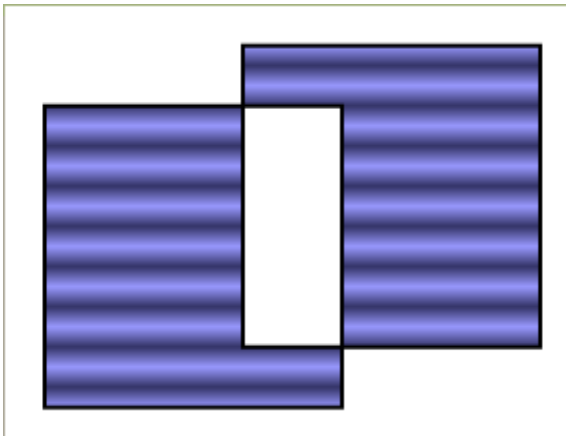
1         <MatrixTransform Matrix=".707,.707,-.707,.707,150,-30" />
2     </LinearGradientBrush.Transform>
3     <LinearGradientBrush.GradientStops>
4         <GradientStop Color="#9999FF" Offset="0.0"/>
5         <GradientStop Color="#333366" Offset="1.0"/>
6     </LinearGradientBrush.GradientStops>
7 </LinearGradientBrush>
8 </Path.Fill>
9 </Path>
    
```

10 This markup is rendered as follows:



11

12 Without the Transform property, this markup would be rendered as follows:

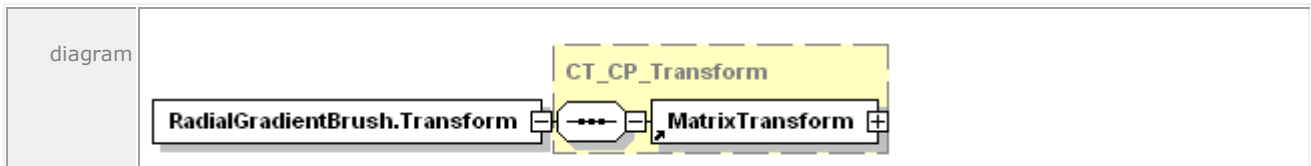


13

14 *end example]*

15 **14.4.9 <RadialGradientBrush.Transform> Element**

16 element **RadialGradientBrush.Transform**



annotation

Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The ellipse defined by the center, gradient origin, x radius, and y radius vaules is transformed using the local effective render transform.

1 *Example 14–26. <RadialGradientBrush.Transform> usage*

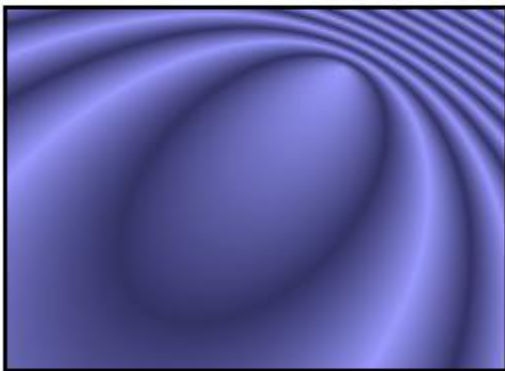
2 The following markup describes a rotation and reposition transform on a radial gradient:

```

3     <Path
4         Stroke="#000000"
5         StrokeThickness="2"
6         Data="M 20,20 L 270,20 L 270,200 L 20,200 Z">
7         <Path.Fill>
8             <RadialGradientBrush
9                 MappingMode="Absolute"
10                Center="80,90"
11                RadiusX="50"
12                RadiusY="80"
13                GradientOrigin="70,15"
14                SpreadMethod="Reflect">
15                <RadialGradientBrush.Transform>
16                    <MatrixTransform Matrix=".707,.707,-.707,.707,150,-10" />
17                </RadialGradientBrush.Transform>
18                <RadialGradientBrush.GradientStops>
19                    <GradientStop Color="#9999FF" Offset="0.0" />
20                    <GradientStop Color="#333366" Offset="1.0" />
21                </RadialGradientBrush.GradientStops>
22            </RadialGradientBrush>
23        </Path.Fill>
24    </Path>

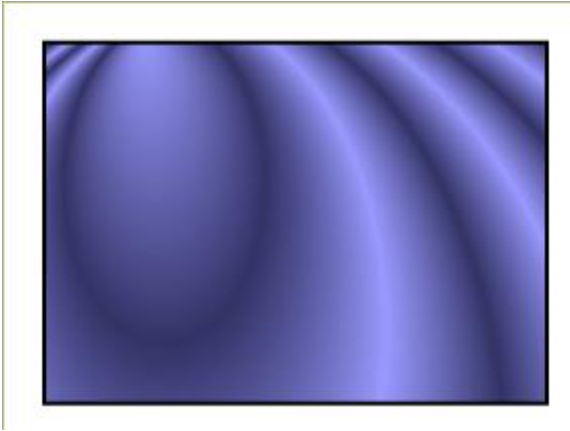
```

25 This markup is rendered as follows:



26

27 Without the Transform property, this markup is rendered as follows:



1

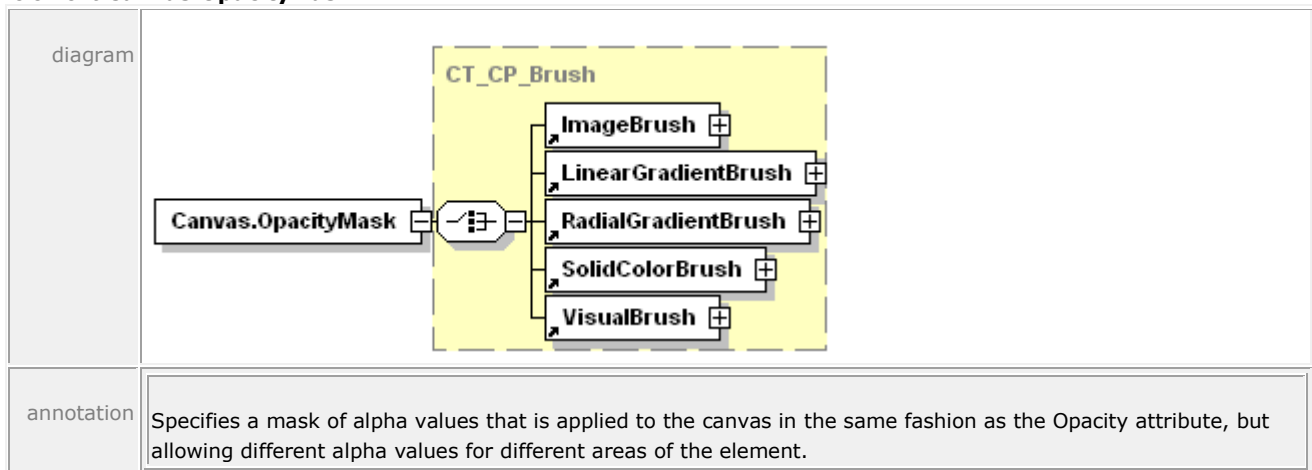
2 *end example]*

1 14.5 OpacityMask

2 The OpacityMask property defines a variable alpha mask for the parent element. The alpha for
3 areas not marked by the brush is 0.0.

4 14.5.1 <Canvas.OpacityMask> Element

5 element **Canvas.OpacityMask**



6 *Example 14–27. <Canvas.OpacityMask> usage*

7 In the following markup, the contents of the canvas are opaque with respect to each other, but
8 both elements are blended with the background triangle:

```

9     <Path Fill="#CCCC66" Data="M 10,10 L 300,80 L 180,240 Z" />
10    <Canvas>
11      <Canvas.OpacityMask>
12        <LinearGradientBrush
13          MappingMode="Absolute"
14          StartPoint="0,150"
15          EndPoint="0,175"
16          SpreadMethod="Pad">
17          <LinearGradientBrush.GradientStops>
18            <GradientStop Color="#40000000" Offset="0.0" />
19            <GradientStop Color="#FF000000" Offset="1.0" />
20          </LinearGradientBrush.GradientStops>
21        </LinearGradientBrush>
22      </Canvas.OpacityMask>
23      <Path
24        Stroke="#000000"
25        StrokeThickness="2"
26        Fill="#333399"
27        Data="M 20,40 L 270,40 L 270,200 L 20,200 Z" />
28      <Glyphs
29        OriginX="30"
30        OriginY="180"
31        UnicodeString="EXAMPLE"
32        FontUri=" ../Resources/Fonts/Timesbd.ttf"

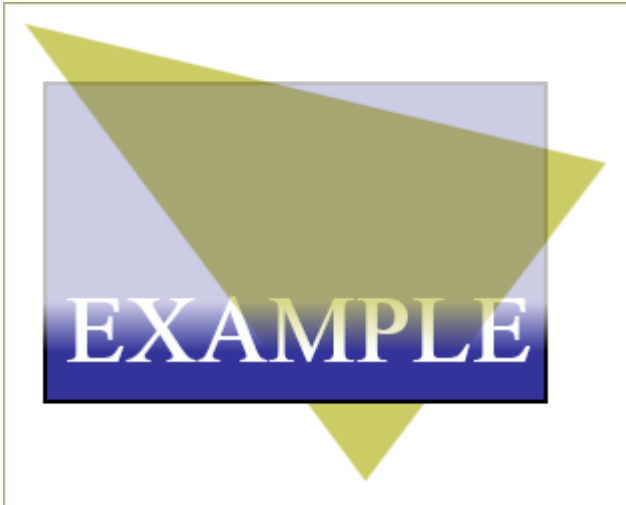
```

```

1      FontRenderingEmSize="48"
2      Fill="#FFFFFF" />
3  </Canvas>

```

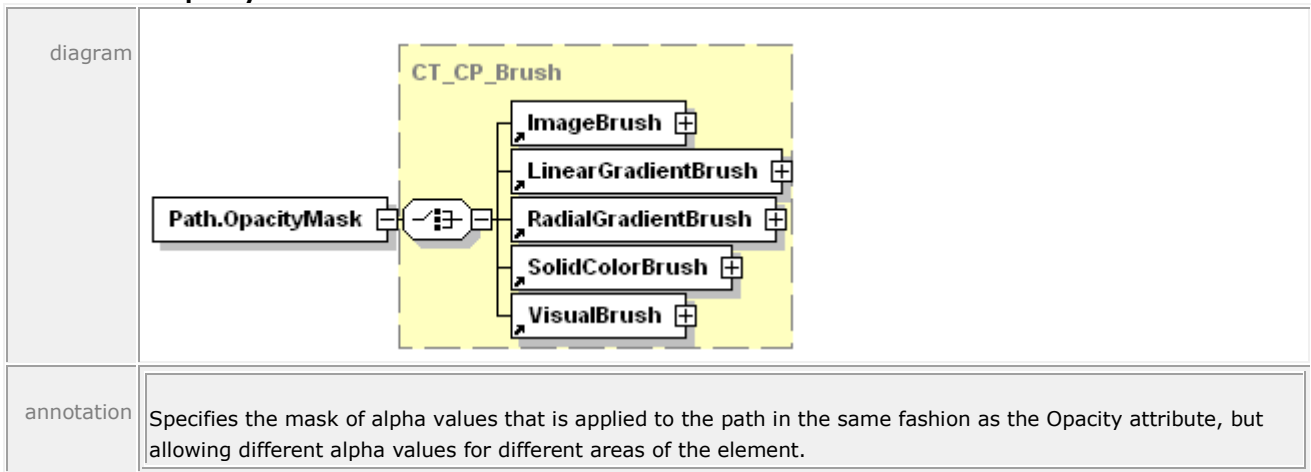
4 This markup is rendered as follows:



5
6 *end example]*

7 **14.5.2 <Path.OpacityMask> Element**

8 element **Path.OpacityMask**



9 *Example 14–28. <Path.OpacityMask> usage*

10 The following markup describes a path that has a linear gradient for the opacity mask and a
11 solid color brush for the fill:

```

12 <Path
13   Stroke="#000000"
14   StrokeThickness="2"
15   Fill="#CCCC66"
16   Data="M 135,10 L 270,250 L 20,250 Z" />
17 <Path
18   Stroke="#000000"

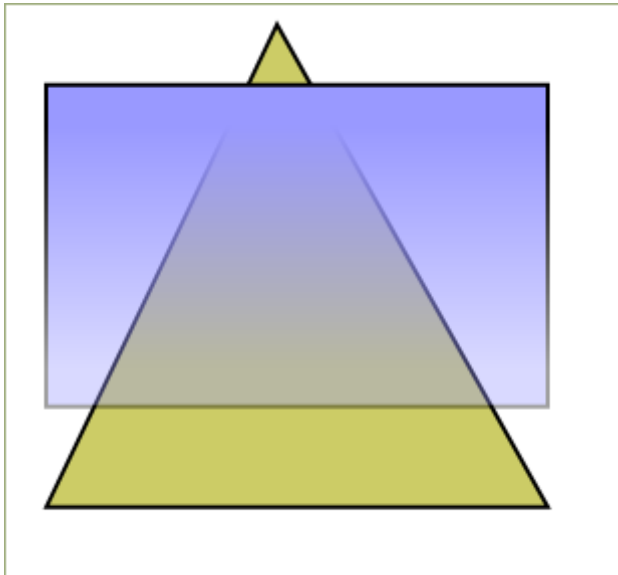
```

```

1      StrokeThickness="2"
2      Data="M 20,40 L 270,40 L 270,200 L 20,200 Z">
3      <Path.OpacityMask>
4          <LinearGradientBrush
5              MappingMode="Absolute"
6              StartPoint="0,60"
7              EndPoint="0,180"
8              SpreadMethod="Pad">
9              <LinearGradientBrush.GradientStops>
10                 <GradientStop Color="#FF000000" Offset="0.0" />
11                 <GradientStop Color="#60000000" Offset="1.0" />
12             </LinearGradientBrush.GradientStops>
13         </LinearGradientBrush>
14     </Path.OpacityMask>
15     <Path.Fill>
16         <SolidColorBrush Color="#9999FF" />
17     </Path.Fill>
18 </Path>

```

19 This markup is rendered as follows:



20
21 *end example]*

22 **14.5.3 <Glyphs.OpacityMask> Element**

23 element **Glyphs.OpacityMask**

<p>diagram</p>	<pre> classDiagram class GlyphsOpacityMask { ~ } class CT_CP_Brush { ImageBrush LinearGradientBrush RadialGradientBrush SolidColorBrush VisualBrush } GlyphsOpacityMask --> CT_CP_Brush </pre> <p>The diagram shows a class <code>Glyphs.OpacityMask</code> with a dashed-line association to a class <code>CT_CP_Brush</code>. The <code>CT_CP_Brush</code> class is highlighted in yellow and contains five subclasses: <code>ImageBrush</code>, <code>LinearGradientBrush</code>, <code>RadialGradientBrush</code>, <code>SolidColorBrush</code>, and <code>VisualBrush</code>. Each subclass has a small icon and a plus sign in its top right corner.</p>
<p>annotation</p>	<p>Specifies a mask of alpha values that is applied to the glyphs in the same fashion as the <code>Opacity</code> attribute, but allowing different alpha values for different areas of the element.</p>

1 *Example 14-29. <Glyphs.OpacityMask> usage*

2 The following markup demonstrates the use of an opacity mask to create a tile effect:

```
3 <Path Fill="#CCCC66" Data="M 40,40 L 480,40 L 260,120 Z" />
4 <Glyphs
5   OriginX="20"
6   OriginY="95"
7   UnicodeString="EXAMPLE"
8   FontUri="../Resources/Fonts/Timesbd.ttf"
9   FontRenderingEmSize="100"
10  Fill="#000080">
11  <Glyphs.OpacityMask>
12    <VisualBrush
13      Viewbox="0,0,2,2"
14      ViewboxUnits="Absolute"
15      Viewport="0,0,6,6"
16      ViewportUnits="Absolute"
17      TileMode="Tile">
18      <VisualBrush.Visual>
19        <Path
20          Fill="#CC000000"
21          Data="M 0,0 L 1.5,0 L 1.5,1.5 L 0,1.5 Z" />
22        </VisualBrush.Visual>
23      </VisualBrush>
24    </Glyphs.OpacityMask>
25  </Glyphs>
```

26 This markup is rendered as follows:



27

28 *end example]*

1 15. Color

2 The mechanisms described in this clause for storing advanced color information in XPS
3 Documents apply to both vector graphics (including text) and raster images. Color producers
4 such as digital cameras and consumers such as printers can store and render significantly more
5 color information than many display devices can render (typically 8 bits per channel). Storing
6 the advanced color information in an XPS Document and passing it through to printing
7 consumers enables greater end-to-end color fidelity.

8 15.1 Color Support

9 XPS Documents support sRGB and ~~rich~~[other](#) color spaces, including scRGB, CMYK, N-Channel,
10 and named colors. Consumers MUST support the following color features:

- 11 • sRGB colors (8 bit-per-channel) in vector data, with and without alpha [M8.1]
- 12 • sRGB colors in image data, using the JPEG, PNG, TIFF, or Windows Media Photo image
13 formats [M8.2]
- 14 • scRGB color specification in vector data, with and without alpha [M8.3]
- 15 • scRGB colors in image data, using the Windows Media Photo image format [M8.4]
- 16 • CMYK colors in vector data [M8.5]
- 17 • CMYK colors in image data, using the TIFF or Windows Media Photo image formats
18 [M8.6]
- 19 • N-Channel colors in vector data [M8.7]
- 20 • N-Channel colors in image data, using the Windows Media Photo image format [M8.8]
- 21 • [profiles compliant with ICC.1:2001-04](#)~~ICC Version 2 profiles~~ for 3-, 4-, 5-, 6-, 7-, and 8-
22 channel color [M8.9]
- 23 • [profiles compliant with ICC.1:2001-04](#)~~ICC Version 2 profiles~~ with a Windows Color
24 System (WCS) profile embedded as a private tag [M8.10]

25 When non-sRGB color information is used, color value specifications are expressed using
26 markup from the XPS Document schema.

27 Consumers are not required to handle all color spaces natively [through every processing stage](#),
28 but, rather, MAY convert data specified in a ~~rich~~ color space [other than sRGB](#) to sRGB at an
29 early stage [O8.1]. Consumers that do not handle [natively rich](#) colors [other than sRGB](#) ~~natively~~
30 can experience reduced fidelity.

31 [The requirements and recommendations of this subclause and its subclauses pertain equally to](#)
32 [raster and vector color content.](#)

33 15.1.1 sRGB Color Space

34 The XPS Document format supports colors in the sRGB color space for both vector and raster
35 graphics.

1 **15.1.2 scRGB Color Space**

2 The XPS Document format supports colors in the scRGB color space for both vector and raster
3 graphics. See §G for the scRGB gamut definition.

4 **15.1.3 Gray Color Space**

5 Gray colors for vector elements can be specified as sRGB or scRGB colors with the red, blue and
6 green components set to the same value. Gray colors for raster images can be specified using
7 any image format.

8 **15.1.4 CMYK Color Space**

9 CMYK color is supported through the use of color management transformations from an ICC
10 profile.

11 **15.1.5 N-Channel Color Spaces**

12 N-channel color is supported through the use of color management transformations from an
13 ICC profile.

14 **15.1.6 Named Color for Spot Colors and N-tone Images**

15 Named colors are supported for images and spot colors through the use of color management
16 transformations from an ICC profile.

17 **15.1.7 Device Color Spaces**

18 To specify colors in the native color space (usually CMYK or N-Channel) for a device, use the
19 standard markup with an ICC profile that approximates the device color space. Include the
20 same ICC device profile in the PageDeviceColorSpaceProfileURI PrintTicket setting.
21 Determination of the color management result for such native color space colors is determined
22 according to mechanisms described in §15.6, Table 15–4. It is the responsibility of the
23 consumer to identify the ICC profile as one that correctly approximates the native colors of the
24 device. When colors in an XPS file are encoded in the native colors of the device, then the
25 colors MUST NOT be color-managed according to the included profile unless forced to do so for
26 transparency effects or gradient blending. [M8.11].

27 **15.1.8 ICC Profiles**

28 XPS Documents MAY include ICC profile parts [O2.3]. XPS producers and consumers MUST
29 provide color management using ICC profiles conforming to the requirements of the ICC Color
30 Profile specification, ICC.1:2001-04 [M8.12], for color spaces other than sRGB and scRGB.
31 Optionally, XPS producers and consumers MAY provide color management using ICC profiles
32 conforming to the requirements of ISO 15076-1, ~~"Image technology colour management —
33 Architecture, profile format, and data structure — Part 1: Based on ICC.1:2004-10"~~ [O8.9].
34 Producers SHOULD restrict ICC profiles to conform to the requirements of the older ICC Color
35 Profile specification, ICC.1:2001-04, when consumer support of the newer ISO version cannot
36 be ascertained [S8.15].

37 Consumers MUST support color profiles as specified in the ICC specification [M8.9]. Supported
38 profiles include Monochrome Input Profiles, Monochrome Display Profiles, Monochrome Output
39 Profiles, Three-component Matrix-based Input Profiles, and RGB Display Profiles. The set of
40 usable N-component LUT-based profiles is limited to 3-, 4-, 5-, 6-, 7-, or 8-color channels.

1 All ICC profiles used in XPS Documents MUST be one of the following [M8.13]:

- 2 • Input
- 3 • Output
- 4 • Monitor (RGB)
- 5 • ColorSpace Conversion

6 One tag is explicitly supported as specified in ICC Version 4.0.0 profiles: the colorant table for
7 named colors. A consumer incapable of supporting named colors SHOULD treat this tag as a
8 user-defined custom tag, and therefore ignore it. It SHOULD instead use the color tables as
9 provided in the profile to convert the specified colors to the Profile Connection Space (PCS)
10 [S8.14].

11 ICC profiles SHOULD be used when embedded in any image format with any color space,
12 except the scRGB color space, for which the gamut boundary described in G is assumed [S8.1].

13 [Note: Some consumers do not correctly apply ICC profiles to grayscale images. *end note*] If
14 consistency of appearance is important, the producer SHOULD adjust the gray tone response
15 curve of the image before adding it to the XPS Document [S8.2].

16 An ICC profile MAY contain the private tag, "MS00", which specifies an embedded Windows
17 Color System (WCS) profile [O8.2].

18 **15.1.9 WcsProfilesTag**

19 The WcsProfilesTag (Windows color space profiles tag) is a private Microsoft ICC profile tag that
20 is used in ICC profiles created by WcsCreateIccProfile to contain input WCS profiles. This tag
21 conforms to ICC profile requirements for profile tags. In particular, the tag header is in big-
22 endian byte ordering, but the embedded WCS XML profiles remain in their native byte order.
23 Furthermore, the tag data must be aligned on a 4-byte boundary (measured from the start of
24 the ICC profile). The structure of the tag is defined by the WcsProfilesTagType below.

25 The WcsProfilesTag signature is "MS00". This is the tag signature that will appear in the ICC
26 profiles tag table for the WcsProfilesTag.

27 *Table 15-1. WcsProfilesTagType structure*

Byte offset	Content
0-3	MS10-type signature.
4-7	Reserved, must be set to 0 (according to ICC convention).
8-11	Byte offset from the beginning of the tag to the CDMP data.
12-15	Size of the CDMP data in bytes.
16-19	Byte offset from the beginning of the tag to the CAMP data.
20-23	Size of the CAMP data in bytes.
24-27	Byte offset from the beginning of the tag to the GMMP data.
28-31	Size of the GMMP data in bytes.
32- <i>n</i>	A sequence of bytes of length <i>element_size-32</i> , where <i>element_size</i> is the tag size recorded in the ICC profile tag table entry for this tag. These are the WCS XML profiles that were used to create this ICC profile via WcsCreateIccProfileFromWcsProfileHandle. The WCS

profiles are ordered as follows: the CDMP (required) first, followed by the CAMP (if present), followed by the GMMP (if present).

1 **15.1.10 WCS Color Profiles**

2 XPS Documents include only WCS color profiles embedded in ICC color profiles, as described
3 above.

4 **15.2 Vector Color Syntax**

5 Vector colors can be specified in XPS Document markup in the following locations:

- 6 • The Color attribute of the <SolidColorBrush> element
- 7 • The Color attribute of the <GradientStop> element
- 8 • The Fill attribute of the <Path> element
- 9 • The Fill attribute of the <Glyphs> element
- 10 • The Stroke attribute of the <Path> element

11 The last three locations are an abbreviated syntax for expressing a solid color brush with the
12 specified color.

1 *Table 15-2. Syntax summary*

Color type	Syntax	Example
sRGB w/o alpha	Color="#RRGGBB"	Color="#FFFFFF"
sRGB with alpha	Color="#AARRGGBB"	Color="#80FFFFFF"
scRGB w/o alpha	Color="sc#RedFloat, GreenFloat,BlueFloat"	Color="sc#1.0,0.5,1.0"
scRGB with alpha	Color="sc#AlphaFloat,RedFloat, GreenFloat,BlueFloat"	Color="sc#0.3,1.0,0.5,1.0"
CMYK with alpha	Color="ContextColor ProfileURI AlphaFloat, Chan0Float, Chan1Float, Chan2Float, Chan3Float"	Color="ContextColor /swopcmypkprofile.icc 1.0,1.0,0.0,0.0,0.0"
N-Channel with alpha	Color="ContextColor ProfileURI AlphaFloat, Chan0Float, ..., ChanN-1Float"	Color="ContextColor /5nchannelprofile.icc 1.0, 1.0, 0.0, 0.0, 1.0, 0.0"
Named color with alpha	Color="ContextColor ProfileURI AlphaFloat, TintFloat, 0, 0"	Color="ContextColor /namedtintprofile.icc 1.0, 1.0, 0, 0"

2 Real numbers specified for color channel values of scRGB and ContextColor colors MUST NOT
3 use exponent forms of numbers [M8.14].

4 It is the responsibility of the consumer to determine profile usability. A consumer incapable of
5 supporting a particular optional ICC profile tag should treat this tag as a user-defined custom
6 tag, and therefore ignore it. If an ICC profile contains optional tags or invalid tag type
7 signatures such that the profile is unusable by a consumer, then the default pixel formats for
8 each vector color space MUST be treated as follows:

- 9 • Three-component integer default for vector data is sRGB.
- 10 • Three-component float default for vector data is scRGB.
- 11 • The specific CMYK to be used as the four component data default for vector data shall be
12 determined by the consumer.

13 A consumer might choose to abort the job.

14 **15.2.1 sRGB Color Syntax**

15 The sRGB color syntax is the same as that used in HTML, with the red, green, and blue
16 channels represented by two hexadecimal digits. XPS Documents can specify an sRGB color
17 either with or without an alpha channel value, which is also expressed as two hexadecimal
18 digits.

19 The syntax is as follows (without alpha):

20 #RRGGBB

21 or (with alpha):

1 #AARRGGBB

2 When an sRGB color is specified without an alpha value, an alpha of "FF" is implied.

3 **15.2.2 scRGB Color Syntax**

4 The scRGB color syntax allows XPS Document producers to specify a color using the full scRGB
5 color space, which is much larger than the sRGB color space and can represent the entire range
6 of colors perceivable by the human eye.

1 This syntax is expressed either as:

2 `sc#RedFloat,GreenFloat,BlueFloat`

3 or:

4 `sc#AlphaFloat,RedFloat,GreenFloat,BlueFloat`

5 When an scRGB color is specified with three numeric values, an alpha of 1.0 is implied. When
6 an scRGB color is specified with four numeric values, the first value is the alpha channel.
7 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be
8 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.15].

9 **15.2.3 CMYK Color Syntax**

10 XPS Document producers specify CMYK colors using the context color syntax, which allows
11 specification of an ICC profile and the individual color channel values as real numbers.

12 The syntax is as follows:

13 `ContextColor ProfileURI AlphaFloat, Chan0Float, Chan1Float, Chan2Float,`
14 `Chan3Float`

15 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST
16 be added as a Required Resource relationship to the FixedPage part [M2.10].

17 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be
18 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.16]. Channel float
19 values MUST also be clamped to the valid range from 0.0 to 1.0 before further processing. If
20 the value is used as input for an ICC profile color transformation, it MUST subsequently be
21 linearly scaled to the range from 0 to 255 or from 0 to 65535, depending on whether the profile
22 uses 8-bit or 16-bit input tables [M8.31].

23 **15.2.4 N-Channel Color Syntax**

24 XPS Document producers specify N-channel colors using the context color syntax, which allows
25 specification of an ICC profile and the individual color channel values as real numbers. The
26 syntax is expressed as follows:

27 `ContextColor ProfileURI AlphaFloat,Chan0Float,...,ChanN-1Float`

28 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST
29 be added as a Required Resource relationship to the FixedPage part [M2.10]. The profile can be
30 a 3-, 4-, 5-, 6-, 7- or 8-channel profile. The context color MUST specify a matching number of
31 channel float values, setting unused ones to 0.0 [M8.17].

32 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be
33 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.18]. Channel float
34 values MUST also be clamped to the valid range from 0.0 to 1.0 before further processing. If
35 the value is used as input for an ICC profile color transformation, it MUST subsequently be
36 linearly scaled to the range from 0 to 255 or from 0 to 65535, depending on whether the profile
37 uses 8-bit or 16-bit input tables [M8.31].

38 To represent a 1-channel profile, create a 3-channel profile and use only the first channel. To
39 represent a 2-channel profile, create a 3-channel profile and use only the first and second
40 channels.

1 **15.2.5 Named Color Syntax**

2 A *named color* is an industry-defined color specification that identifies a particular color in a
 3 well-defined color schema, usually for the purpose of printing. There are currently several
 4 named color schemas. Producers specify named colors using the context color syntax, which
 5 allows specification of a named color using an ICC profile. The named color schema used is
 6 determined by the ICC profile.

7 A named color is expressed as a combination of an ink name stored in the ICC profile and a tint
 8 level (percentage ink dilution). The ink name for the tint is contained in the colorantTable clrt
 9 tag. This tag is not defined for ~~ICC Version 3.4 profiles~~ [profiles compliant with ICC.1:2001-04](#),
 10 and its presence is benignly ignored by ICM V2 implementations. Therefore, it is used in XPS
 11 Documents to specify the names of named colors.

12 The syntax for referencing a single named color is as follows:

```
13 ContextColor ProfileURI AlphaFloat,TintFloat,0,0
```

14 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST
 15 be added as a Required Resource relationship to the FixedPage part [M2.10]. The profile MUST
 16 have 3, 4, 5, 6, 7 or 8 channels (and an *n*CLR signature, where *n* is the number of channels),
 17 mapping to a valid PCS [M8.19]. To represent a single tone or duotone profile, create a 3-
 18 channel profile and specify a colorant name in the profile's colorantTable with zero-length
 19 (name field all set to 0) for the unused channels.

20 `AlphaFloat` specifies the alpha to be applied to the named color. `TintFloat` specifies how diluted
 21 with respect to the color schema's white color point the named color is, with 1.0 being the pure
 22 named color and 0.0 being fully diluted. The final two values are always set to 0 to specify a
 23 single named color.

24 For duotone named colors, use the first two values and set the final value to 0. [*Example*: The
 25 syntax for referencing a duotone named color is as follows:

```
26 ContextColor ProfileURI AlphaFloat,Tint0Float,Tint1Float,0
```

27 *end example*]

28 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be
 29 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.20]. The tint float
 30 value MUST also be clamped to the valid range from 0.0 to 1.0 before further processing. If the
 31 value is used as input for an ICC profile color transformation, it MUST subsequently be linearly
 32 scaled to the range from 0 to 255 or from 0 to 65535, depending on whether the profile uses 8-
 33 bit or 16-bit input tables [M8.31].

34 Consumers that do not understand the named color MUST compute a color approximation
 35 through ICM-compliant color management functions using the specified profile. When a named
 36 color is used in a gradient brush or with transparency, the result produced by consumers that
 37 are not named-color aware MAY differ significantly from the result produced by consumers that
 38 are named-color aware [O8.3].

39 An XPS consumer that is aware of named colors looks for the clrt tag to find out if a specific
 40 ContextColor designates a named color description. A producer of XPS documents containing
 41 named colors SHOULD create the color profile in such a way that a linear ramp of the channel
 42 values corresponding to a named colorant maps to PCS values resulting in the same color
 43 appearance for consumers unaware of named colors (or the specific colorant) [S8.3].

1 This method can be used for 1 to 8 named colors; however, the ContextColor syntax requires a
2 minimum of 1 Alpha value and 3 Channel values. It is RECOMMENDED that a 1 or 2 tone profile
3 uses the first 1 or 2 channels, respectively, and specifies 0 for the remaining channels [S8.4].

4 If the consumer does not know ALL of the colorants named in the clr tag, it SHOULD treat the
5 profile as if it were a regular N-channel source profile and SHOULD NOT attempt to use any of
6 the known colorants, as that would result in undefined results [S8.6].

7 A named color profile MAY be used with images [O8.4], which is especially useful for the
8 reproduction of corporate logo images using strictly controlled ink sets. A named profile could
9 contain SWOP CMYK in the first 4 channels, and use the remaining 4 channels for highlight
10 colors. This allows the use of up to 4 accent colors in images.

11 **15.3 Rich-Colors in Raster Images**

12 This subclause describes specific considerations for including [color](#) raster images ~~with rich colors~~
13 in XPS Documents.

14 **15.3.1 sRGB Raster Images**

15 XPS Documents support sRGB raster images in the following formats:

- 16 • JPEG
- 17 • PNG
- 18 • TIFF
- 19 • Windows Media Photo

20 The following Windows Media Photo pixel formats are supported:

- 21 • WICPixelFormat24bppRGB
- 22 • WICPixelFormat24bppBGR
- 23 • WICPixelFormat32bppBGR
- 24 • WICPixelFormat32bppBGRA
- 25 • WICPixelFormat32bppPBGRA
- 26 • WICPixelFormat48bppRGB
- 27 • WICPixelFormat64bppRGBA
- 28 • WICPixelFormat64bppPRGBA

29 Pixel formats WICPixelFormat32bppPBGRA and WICPixelFormat64bppPRGBA are pre-multiplied
30 alpha formats. See §18.4.1 for details.

31 The following Windows Media Photo packed pixel formats are supported:

- 32 • WICPixelFormat16bppBGR555
- 33 • WICPixelFormat16bppBGR565
- 34 • WICPixelFormat32bppBGR101010

35 See §9.1.5 for more details.

1 **15.3.2 scRGB Raster Images**

2 XPS Documents support scRGB raster images only in the Windows Media Photo image format.
3 The following pixel formats are supported:

- 4 • WICPixelFormat48bppRGBFixedPoint
- 5 • WICPixelFormat48bppRGBHalf
- 6 • WICPixelFormat96bppRGBFixedPoint
- 7 • WICPixelFormat128bppRGBFloat
- 8 • WICPixelFormat64bppRGBAFixedPoint
- 9 • WICPixelFormat64bppRGBFixedPoint
- 10 • WICPixelFormat64bppRGBHalf
- 11 • WICPixelFormat64bppRGBHalf
- 12 • WICPixelFormat128bppRGBAFixedPoint
- 13 • WICPixelFormat128bppRGBFixedPoint
- 14 • WICPixelFormat128bppRGBAFloat
- 15 • WICPixelFormat128bppPRGBAFloat
- 16 • WICPixelFormat32bppRGBE

17 Pixel format WICPixelFormat128bppPRGBAFloat is a pre-multiplied alpha format. See §18.4.1
18 for details.

19 **15.3.3 Gray Raster Images**

20 XPS Documents support gray raster images in the following formats:

- 21 • JPEG
- 22 • PNG
- 23 • TIFF
- 24 • Windows Media Photo

25 The following Windows Media Photo pixel formats are supported:

- 26 • WICPixelFormatBlackWhite
- 27 • WICPixelFormat8bppGray
- 28 • WICPixelFormat16bppGray
- 29 • WICPixelFormat16bppGrayFixedPoint (scRGB range)
- 30 • WICPixelFormat16bppGrayHalf (scRGB range)
- 31 • WICPixelFormat32bppGrayFixedPoint (scRGB range)
- 32 • WICPixelFormat32bppGrayFloat

33 **15.3.4 CMYK Raster Images**

34 | CMYK images are stored in TIFF-~~6.0~~ or Windows Media Photo format.

1 **15.3.4.1 TIFF CMYK Raster Images**

2 CMYK TIFF image tags are described in §9.1.5.3.

3 ICC profiles can be associated with CMYK raster images by using an ICC profile embedded in
4 the TIFF file (tag 34675) or associated using the mechanism described in §15.3.8.

5 **15.3.4.2 Windows Media Photo CMYK Raster Images**

6 The Windows Media Photo CMYK format is described in the Windows Media Photo specification.
7 The following formats are supported:

- 8 • WICPixelFormat32bppCMYK
- 9 • WICPixelFormat40bppCMYKAlpha
- 10 • WICPixelFormat64bppCMYK
- 11 • WICPixelFormat80bppCMYKAlpha

12 **15.3.4.3 JPEG CMYK Raster Images**

13 Support for JPEG CMYK images varies by implementation and SHOULD NOT be used in XPS
14 Documents [S2.7]. See §9.1.5.1 for more details.

15 **15.3.5 N-channel Raster Images**

16 N-channel images are stored in the Windows Media Photo image file format using an ICC
17 profile. The following formats are supported:

- 18 • WICPixelFormat24bpp3Channels, WICPixelFormat48bpp3Channels
- 19 • WICPixelFormat32bpp4Channels, WICPixelFormat64bpp4Channels
- 20 • WICPixelFormat40bpp5Channels, WICPixelFormat80bpp5Channels
- 21 • WICPixelFormat48bpp6Channels, WICPixelFormat96bpp6Channels
- 22 • WICPixelFormat56bpp7Channels, WICPixelFormat112bpp7Channels
- 23 • WICPixelFormat64bpp8Channels, WICPixelFormat128bpp8Channels
- 24 • WICPixelFormat32bpp3ChannelsAlpha, WICPixelFormat64bpp3ChannelsAlpha
- 25 • WICPixelFormat40bpp4ChannelsAlpha, WICPixelFormat80bpp4ChannelsAlpha
- 26 • WICPixelFormat48bpp5ChannelsAlpha, WICPixelFormat96bpp5ChannelsAlpha
- 27 • WICPixelFormat56bpp6ChannelsAlpha, WICPixelFormat112bpp6ChannelsAlpha
- 28 • WICPixelFormat64bpp7ChannelsAlpha, WICPixelFormat128bpp7ChannelsAlpha
- 29 • WICPixelFormat72bpp8ChannelsAlpha, WICPixelFormat144bpp8ChannelsAlpha

30 **15.3.6 Named Color Raster Images**

31 Named color (N-tone) raster images are stored in the Windows Media Photo image file format
32 using an ICC profile that maps the tint channel combinations to valid PCS values. See §15.3.5
33 for pixel format definitions.

34 Consumers unaware of named colors can then compute color approximations using the PCS
35 values computed from the profile.

1 **15.3.7 Device Color Raster Images**

2 Device color (N-channel) raster images are stored in the Windows Media Photo image file
3 format in the same manner as a named color raster image. See§15.1.7 for more details. RGB
4 and CMYK raster images can also be stored in the TIFF image file format. JPEG CMYK images
5 SHOULD NOT be used [S2.7].

6 **15.3.8 Images and Color Profile Association**

7 Images can use a color profile matching the channel configuration of the image using one of
8 two methods:

- 9 • Color profile embedded in an image using the image format specific mechanism
- 10 • Color profile contained in a separate part associated with the image using the following
11 markup:

```
12 <ImageBrush ImageSource="{ColorConvertedBitmap image.tif profile.icc}" ... />
```

13 An associated color profile overrides an embedded color profile and is processed instead
14 of any embedded color profile. The profile URI MUST be added as a Required Resource
15 relationship to the FixedPage part [M2.10].

16 **15.3.9 Color Space Pixel Formats for Raster Images**

17 It is the responsibility of the consumer to determine profile usability. A consumer incapable of
18 supporting a particular optional ICC profile tag should treat this tag as a user-defined custom
19 tag, and therefore ignore it. If an ICC profile is not embedded or associated with a raster image
20 or if both the embedded and associated profiles, which are present, are not compatible with the
21 pixel format of the image or contain optional tags or invalid tag type signatures such that
22 neither profile is usable by a consumer, then the default pixel formats for each raster color
23 space MUST be treated as follows [M8.30].

24 *Table 15–3. Color Space Pixel Format Defaults*

Color Space	Pixel Formats
sRGB	Integer RGB Integer Grayscale Integer 3-Channel
scRGB (wcsRGB gamut)	Floating Point scRGB Half-Float scRGB Fixed-Point scRGB Floating Point Grayscale Half-Float Grayscale Fixed-Point Grayscale
CMYK (with SWOP profile)	Integer CMYK Integer 4-Channel Integer 5-Channel (ignore channel 5) Integer 6-Channel (ignore channels 5 and 6) Integer 7-Channel (ignore channels 5, 6, and 7) Integer 8-Channel (ignore channels 5, 6, 7, and 8)

15.4 Color Separation

Consumers MAY perform color separation, if desired [O8.5].

A named color used for markings that are intended to be rendered on every layer of the separation can be specified with the DocumentImpositionColor PrintTicket setting.

The color name specified by the DocumentImpositionColor PrintTicket setting MUST be matched only to profiles containing exactly one non-zero-length colorant name in the profile's colorantTable [M8.22]. The color name specified by the DocumentImpositionColor setting serves as a label for that color only and MUST NOT be matched against any Named Colors known by the consumer [M8.23]. The comparison of the color name specified by the DocumentImpositionColor PrintTicket setting with the colorant name in the profile's colorantTable MUST be performed as a case-sensitive ASCII comparison after trimming leading and trailing whitespace from each string [M8.24].

The imposition named color is used *only* to compute XYZ values for consumers that do not perform separation. For consumers that do perform separation, it is an indicator that the tint level supplied SHOULD be used for all device colorants [S8.7]. Producers SHOULD create the profile used by the imposition color in such a way that it does not lay down excessive ink when printed on a device that does not perform separation and uses the profile to compute XYZ values instead [S8.8].

15.5 Alpha and Gradient Blending ~~with Rich Colors~~

For consumers that ~~understand rich~~ handle colors other than sRGB, it is necessary to understand how they can be blended to create gradient or transparency effects. The PrintTicket specifies the color space that SHOULD be used for blending gradients and transparencies in the PageBlendColorSpace setting [S8.9]. These settings apply to the page level.

If a consumer understands the PageBlendColorSpace PrintTicket setting, it SHOULD convert all color to the specified blending color space before performing a blend operation [S8.9]. For gradients, the specified blending color space is used only if no gradient stop color values are specified using sRGB or scRGB colors. If any of the gradient stop color values are specified using sRGB or scRGB colors or the consumer does not understand the PageBlendColorSpace PrintTicket setting, the color interpolation mode of the gradient brush MUST be used instead [M8.25].

~~The behavior of documents using rich color features is implementation specific.~~ Consumers MUST support sRGB [M8.1], but they MAY support alpha and gradient blending with other rich color spaces such as scRGB or CMYK ~~as well~~ [O8.6]. The behavior of documents using non-sRGB alpha and gradient blending is implementation specific. Consumers that encounter any document using non-sRGB colors MAY process those colors using conversion to the simpler sRGB color space, resulting in deviations, especially for alpha blending [O8.6].

1 15.6 PrintTicket Color Settings

2 This subclause summarizes the color-related PrintTicket settings. For more information, refer to
3 the Print Schema.

4 *Table 15-4. PrintTicket color settings*

Feature or ParameterDef	Option/ScoredProperty or Properties	Description
PageColorManagement	Device (default)	Perform color management only in device.
	Driver	Allow driver to perform color management. The driver MAY color manage elements or convert them to different color spaces [O8.7].
PageDeviceColorSpaceProfileURI	<i>profileUri</i> properties	Identifies an ICC profile contained in the XPS package. The processing of this option depends of the setting of the PageDeviceColorSpaceUsage feature. No default value is specified. Contains an absolute part name relative to the package root. All elements using that profile are assumed to be already in the appropriate device color space, and will not be color managed in the driver or device.
PageDeviceColorSpaceUsage	MatchToDefault (default)	If the device determines that the profile specified by the PageDeviceColorSpaceProfileURI feature can be used as a device color space profile, all elements using the same profile are treated as already being specified in device color space. However, the device's internal color profile SHOULD be used for color management of all other elements [S8.10].

		If the profile cannot be used as a device color space profile, elements using the profile MUST be color managed like any other element using a color profile [M8.27].
	OverrideDeviceDefault	If the profile specified by the PageDeviceColorSpaceProfileURI parameter definition has a number of channels matching the number of primaries of the device, it SHOULD be used instead of the device's internal color management for all elements [S8.11]. Elements using this profile are assumed to be in device color space and will not be color managed further.
PageBlendColorSpace	sRGB (default)	The sRGB color space that SHOULD be used for blending [S8.9].
	scRGB	The scRGB color space that SHOULD be used for blending [S8.9].
	ICCProfile	The Uri property of the option specifies an ICC profile defining the color space that SHOULD be used for blending [S11.13]. The Uri is an absolute part name relative to the package root. The profile MUST be an output profile (containing AToB0Tag, BToA0Tag, AToB1Tag, BToA1Tag, AToB2Tag, and BToA2Tag), otherwise it MUST be ignored [M8.28]. The rendering intent specified by PageICMRenderingIntentPrintTicket setting is used, unless the profile specifies a rendering intent of its own.

		Elements using the profile specified by PageBlendColorSpace MAY be blended naively (channel-by-channel) without converting through PCS [O8.8].
PageICMRenderingIntent	AbsoluteColorimetric RelativeColorimetric (default) Photographs BusinessGraphics	The rendering intent as defined by profiles compliant with ICC.1:2001-04 the ICC Version 2 specification . This value SHOULD be ignored for elements using an ICC profile that specifies the rendering intent in the profile [S8.13].
DocumentImpositionColor	<i>colorName properties</i>	Elements using the named color identified by the colorName properties MUST appear on all color separations [M8.29]. See §15.4 and 15.2.5 for details. Consumers that do not produce separations treat these elements like other elements using named color, without any additional required processing steps. No default value is specified.
PageBlackGenerationProcessing	Automatic	Default.
	Custom/TotalInkCoverageLimit	The maximum allowable sum of the four ink coverages anywhere in an image or element (200% to 400%).
	Custom/BlackInkLimit	The maximum allowed K-channel value (0% to 100%).
	Custom/GrayComponentReplacementLevel	The percentage of gray component replacement to perform (0% to 100%).
	Custom/GrayComponentReplacementStart	The point in the highlight-to-shadow range where GCR should start (0% to 100%; 100% = darkest

	shadow).
Custom/ GrayComponentReplacement Extent	The extent beyond neutrals (into chromatic colors) that GCR applies. 0% = Undercolor component replacement; 100% = Gray component replacement.
Custom/UnderColorAdditionS tart	The shadow level below which UCA will be applied (0% to 100%).
Custom/UnderColorAdditionL evel	The amount of chromatic ink (in gray component ratios) to add to areas where GCR/UCR has generated "BlackInkLimit" (or "UCAStart", if specified) in the dark neutrals and near-neutral areas (0% to 100%).

1

2

16. Document Structure and Interactivity

Some consumers support enhanced interactive functionality through features such as text selection, navigation, and hyperlinking. Others, such as screen readers, provide enhanced accessibility. These features rely on structural information beyond what can be inferred from the page markup. Producers can author this information explicitly.

The methods for adding document structure described here are OPTIONAL [O9.1]. Consumers MAY ignore any authored document structure or hyperlinks [O9.1], particularly where they are not relevant (such as in the case of printers). Recommended consumer behavior in the absence of document structure information is also described.

Document structure is defined with markup in the FixedPage, FixedDocument, DocumentStructure, and StoryFragments parts.

16.1 Document Structure Markup

Document structure markup consists of two structural concepts. The first is the *document outline*, which contains a structured list of indices into the XPS Document, similar to a table of contents. The second is the *document content*, which identifies blocks of individually readable content. Each of these blocks is called a *story*.

A story can extend across multiple pages, and several stories can share a single page. A story can include the entire contents of an XPS Document, or it can include only an individual block of readable content, such as a single newspaper article. Like a newspaper article, the story can appear in blocks throughout the XPS Document. [*Example*: The first part could appear on page 1 and the second part on page 5. *end example*] Since a story can span multiple pages, the document content identifies which FixedPage parts contain fragments of a particular story.

A *story fragment* is the portion of a story that appears within a single fixed page. The story fragment contains the structural markup for all text and images related to a particular story on a particular page. When a producer specifies the document structure, every FixedPage part has a corresponding StoryFragments part that contains all of the story fragments for that page.

Each story fragment contains content structure information. *Content structure* is the set of markup elements that allow expression of well-understood semantic blocks, such as paragraphs, tables, lists, and figures. Content structure markup enables features such as paragraph and table selection, screen reading, and rich-format copying.

Producers MAY provide either the document outline or the document content, or both; consumers MAY ignore either or both [O9.2].

16.1.1 DocumentStructure Part

The fundamental building block of document structure markup is the named element. A *named element* refers to an element in the fixed page markup with a specified Name attribute. Every meaningful element in the fixed page markup SHOULD specify a Name attribute in order for the document structure markup to refer to it [S9.1].

Document structure markup SHOULD NOT refer to a single named element more than once in the document content or to a named element that embeds another named element that it also

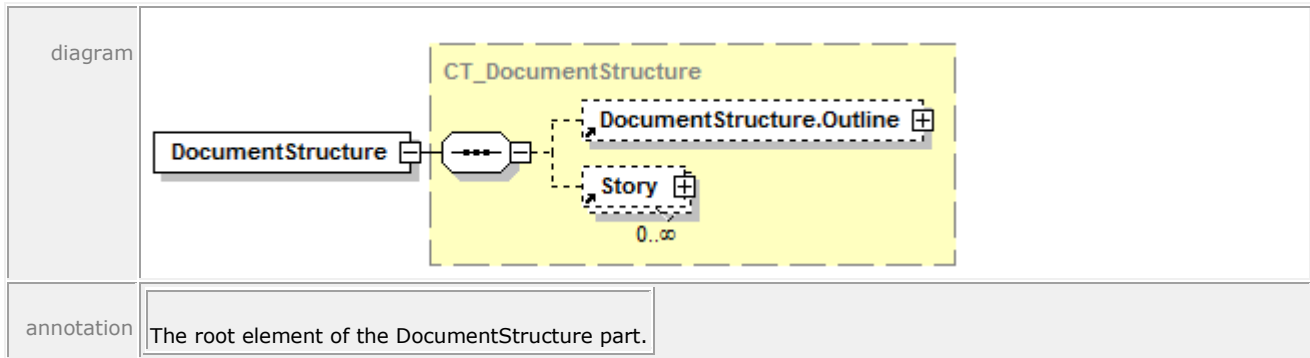
1 refers to. When referring to a <Canvas> element, producers SHOULD consider all descendant
 2 elements to be referenced in markup order [S9.3]. Consumers MAY choose to interpret these
 3 scenarios as duplicate document content [O9.3].

4 Children of <VisualBrush> elements SHOULD NOT be referenced by document structure
 5 markup [S9.30].

6 Because each named element in a FixedPage part that is intended as an addressable location is
 7 specified in the <PageContent.LinkTargets> element in the FixedDocument part, consumers
 8 MAY first attempt to locate named elements directly from the FixedDocument part [O9.4].

9 **16.1.1.1 <DocumentStructure> Element**

10 element **DocumentStructure**



11 The <DocumentStructure> element is the root element of the DocumentStructure part. It MAY
 12 contain a single <DocumentStructure.Outline> element and zero or more <Story> elements
 13 [M2.72].

14 *Example 16–1. Document structure markup*

```

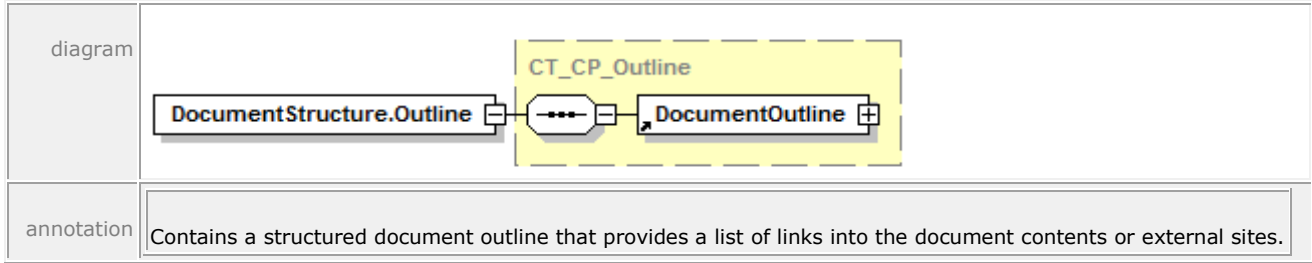
15 <DocumentStructure
16   xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
17   <DocumentStructure.Outline>
18     ...
19   </DocumentStructure.Outline>
20   <Story>
21     ...
22   </Story>
23   <Story>
24     ...
25   </Story>
26 </DocumentStructure>

```

27 *end example]*

1 **16.1.1.2 <DocumentStructure.Outline> Element**

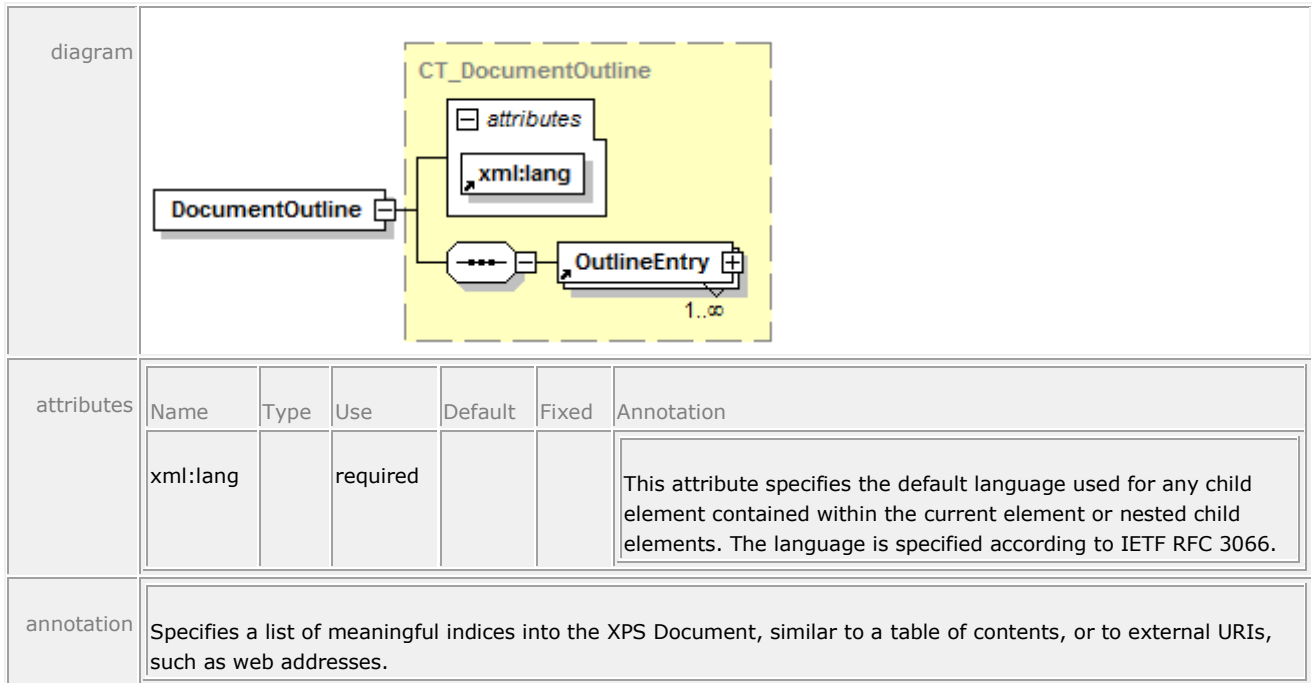
2 element **DocumentStructure.Outline**



3 The <DocumentStructure.Outline> element is the root element of the document outline. The
 4 <DocumentStructure.Outline> element contains only a single <DocumentOutline> element.

5 **16.1.1.3 <DocumentOutline> Element**

6 element **DocumentOutline**



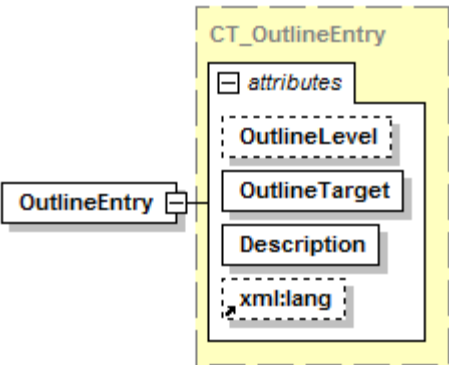
7 The <DocumentOutline> element lets producers specify an organizational hierarchy in the form
 8 of a list of URIs to locations in the fixed page markup or to external addresses, similar to a
 9 table of contents or a set of bookmarks. The <DocumentOutline> element contains only
 10 <OutlineEntry> elements.

11 The xml:lang attribute specifies the default language used by the Description attribute of the child
 12 <OutlineEntry> element.

13 Consumers can use the document outline to implement such features as a table of contents or
 14 a navigation pane.

15 **16.1.1.4 <OutlineEntry> Element**

16 element **OutlineEntry**

<p>diagram</p>																																				
<p>attributes</p>	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>OutlineLevel</td> <td>ST_IntGEOne</td> <td>optional</td> <td>1</td> <td></td> <td>A description of the level where the outline entry exists in the hierarchy. A value of 1 is the root.</td> </tr> <tr> <td>OutlineTarget</td> <td>xs:anyURI</td> <td>required</td> <td></td> <td></td> <td>The URI to which the outline entry is linked. This can be a URI to a named element within the document or an external URI, such as a website. It can be used as a hyperlink destination.</td> </tr> <tr> <td>Description</td> <td>xs:string</td> <td>required</td> <td></td> <td></td> <td>The friendly text associated with this outline entry.</td> </tr> <tr> <td>xml:lang</td> <td></td> <td>optional</td> <td></td> <td></td> <td>This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	OutlineLevel	ST_IntGEOne	optional	1		A description of the level where the outline entry exists in the hierarchy. A value of 1 is the root.	OutlineTarget	xs:anyURI	required			The URI to which the outline entry is linked. This can be a URI to a named element within the document or an external URI, such as a website. It can be used as a hyperlink destination.	Description	xs:string	required			The friendly text associated with this outline entry.	xml:lang		optional			This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.					
Name	Type	Use	Default	Fixed	Annotation																															
OutlineLevel	ST_IntGEOne	optional	1		A description of the level where the outline entry exists in the hierarchy. A value of 1 is the root.																															
OutlineTarget	xs:anyURI	required			The URI to which the outline entry is linked. This can be a URI to a named element within the document or an external URI, such as a website. It can be used as a hyperlink destination.																															
Description	xs:string	required			The friendly text associated with this outline entry.																															
xml:lang		optional			This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.																															
<p>annotation</p>	<p>Represents an index to a specific location in the document.</p>																																			

1 Each <OutlineEntry> element represents an index to a specific location in the document or a
 2 specific location external to the document. Consumers can use the document outline
 3 information to support interactive functionality.

4 *Example 16–2. Document outline markup*

5 A viewing consumer can create a navigation pane that uses the Unicode value of the Description
 6 attribute of each <OutlineEntry> element. The corresponding location is specified by the
 7 OutlineTarget attribute, which are specified in a manner identical to hyperlinks. The OutlineLevel
 8 attribute allows consumers to indent entries in the navigation pane.

```

9     <DocumentStructure
10         xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
11         <DocumentStructure.Outline>
12             <DocumentOutline>
13                 <OutlineEntry
14                     OutlineLevel="1"
15                     Description="1. Documents"
    
```



```

1         OutlineTarget="../FixedDoc.fdoc#Documents_1" />
2     <OutlineEntry
3         OutlineLevel="2"
4         Description="1.1. Paragraphs"
5         OutlineTarget="../FixedDoc.fdoc#Paragraphs_1_1" />
6     </DocumentOutline>
7 </DocumentStructure.Outline>
8 </DocumentStructure>

```

9 A consumer might display this information as follows, with the first entry linked to Documents_1
10 and the second entry linked to Paragraphs_1_1.

- 11 1. Documents
- 12 1.1. Paragraphs

13 *end example]*

14 **16.1.1.5 <Story> Element**

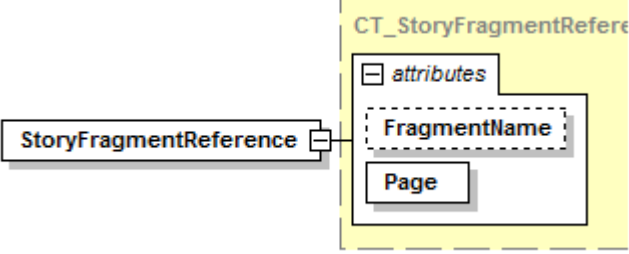
15 element **Story**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	StoryName	xs:string	required			The name used by story fragments to identify they belong to this story.
annotation	Defines a single story and where each of its story fragments appear in the XPS Document.					

16 The <Story> element is the root for a single story and orders all of the story fragments
17 containing content structure information such as sections, paragraphs, and tables. Each story
18 has a unique name that is used to correlate the content structure for each page to that story.
19 The <Story> element contains one or more <StoryFragmentReference> elements.

1 **16.1.1.6 <StoryFragmentReference> Element**

2 element **StoryFragmentReference**

diagram	 <p>The diagram shows a box labeled 'StoryFragmentReference' connected to a larger box titled 'CT_StoryFragmentReferenc'. Inside this larger box, there are two sub-boxes: 'FragmentName' (indicated as an attribute with a dashed border) and 'Page' (indicated as an attribute with a solid border). A small box labeled 'attributes' is also present at the top of the larger box.</p>																							
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>FragmentName</td> <td>xs:string</td> <td>optional</td> <td></td> <td></td> <td>Used to distinguish between multiple story fragments from the same story on a single page.</td> </tr> <tr> <td>Page</td> <td>ST_IntGEOne</td> <td>required</td> <td></td> <td></td> <td>Identifies the page number of the document that the story fragment is related to. Page numbers start at 1 and correspond to the order of <PageContent> elements in the FixedDocument part.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	FragmentName	xs:string	optional			Used to distinguish between multiple story fragments from the same story on a single page.	Page	ST_IntGEOne	required			Identifies the page number of the document that the story fragment is related to. Page numbers start at 1 and correspond to the order of <PageContent> elements in the FixedDocument part.					
Name	Type	Use	Default	Fixed	Annotation																			
FragmentName	xs:string	optional			Used to distinguish between multiple story fragments from the same story on a single page.																			
Page	ST_IntGEOne	required			Identifies the page number of the document that the story fragment is related to. Page numbers start at 1 and correspond to the order of <PageContent> elements in the FixedDocument part.																			
annotation	<p>Identifies the StoryFragments part where this individual story fragment is defined.</p>																							

3 The <StoryFragmentReference> element identifies the page with a relationship to the
 4 StoryFragments part in which the single story fragment is defined. By identifying where in the
 5 XPS Document each story fragment appears, consumers can easily access only the pages that
 6 contain a particular story.

7 Each page that contains a story fragment is identified by number. This number refers to the *n*th
 8 page of the XPS Document referenced within the fixed document sequence and fixed document
 9 markup, starting at the fixed payload root. This value is identified in the Page attribute. The
 10 StoryFragments part containing the corresponding content structure is referenced by retrieving
 11 the part associated via relationship from the indicated page. This allows consumers to access
 12 only the pages of the document that contain the story of interest. It is also possible for a single
 13 story to return to a page containing a different fragment of the same story.

14 The FragmentName attribute MUST be unique within the scope of the story [M2.72].

1 *Example 16-3. Simple multi-story document*

2 The following markup describes a four-page document containing one story that covers the first
3 one and one-half pages and then continues on page 4. It is interrupted by a second story that
4 begins in the middle of page 2 and concludes on page 3.

```
5     <DocumentStructure
6         xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
7         <Story Name="Story1">
8             <StoryFragmentReference Page="1"/>
9             <StoryFragmentReference Page="2"/>
10            <StoryFragmentReference Page="4"/>
11        </Story>
12        <Story Name="Story2">
13            <StoryFragmentReference Page="2"/>
14            <StoryFragmentReference Page="3"/>
15        </Story>
16    </DocumentStructure>
```

17 *end example]*

18 *Example 16-4. Story flowing back and forth across a page boundary*

19 The following markup describes a page containing two tables, arranged side-by-side, each of
20 which continues to the following page. In this case, the fragment is split and a fragment name
21 is specified. `FragmentA` refers to the content leading up to the middle of the first (left) table and
22 `FragmentB` is the continuation of this table on the following page. The flow then returns to the
23 second (right) table on page 1 (`FragmentC`) before continuing with the rest of the story in
24 `FragmentD`.

```
25     <DocumentStructure
26         xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
27         <Story Name="Story_1">
28             <StoryFragmentReference FragmentName="FragmentA" Page="1"/>
29             <StoryFragmentReference FragmentName="FragmentB" Page="2"/>
30             <StoryFragmentReference FragmentName="FragmentC" Page="1"/>
31             <StoryFragmentReference FragmentName="FragmentD" Page="2"/>
32         </Story>
33     </DocumentStructure>
```

34 *end example]*

35 **16.1.2 StoryFragments Part**

36 The StoryFragments part contains content structure markup (describing such things as tables
37 and paragraphs) for each story fragment that appears on the page. The content structure is
38 expressed by tags that ultimately wrap `<NamedElement>` references that point to fixed page
39 markup.

40 *Table 16-1. StoryFragments part elements*

Name	Description
<code><StoryFragments></code>	Root element.
<code><StoryFragment></code>	Contains all content structure markup elements for a single story fragment.

<StoryBreak>	Presence of this element indicates that the following or preceding markup is not continued to the previous or next story fragment, depending on whether the element is at the beginning or end of the story fragments markup.
<SectionStructure>	Arbitrary structural grouping element.
<TableStructure>	Contains a full table definition.
<TableRowGroupStructure>	Contains a group of table rows.
<TableRowStructure>	Contains a row of table cells.
<TableCellStructure>	Contains structural elements representing the contents of a table cell.
<ListStructure>	Group of related items.
<ListItemStructure>	Individual item in a list.
<FigureStructure>	Group of related named elements that should be interpreted as a whole (such as a diagram).
<ParagraphStructure>	Group of named elements that constitute a paragraph.
<NamedElement>	Element that links the document structure markup to the fixed page markup.

1 Because a single content structural element can be split across pages, the <StoryBreak>
 2 element is provided to identify that a given element continues *to* the next story fragment or
 3 continues *from* a previous story fragment. A <StoryBreak> element MUST NOT be included in a
 4 position other than the first or last child element of a <StoryFragment> element [M2.72].

5 If a <StoryBreak> element is not present at the beginning of the content structure markup,
 6 consumers SHOULD consider the markup a continuation of the previous story fragment that
 7 must be merged [S9.4]. Likewise, if a <StoryBreak> element is not present at the end of the
 8 content structure markup, consumers SHOULD consider the markup a continuation to the next
 9 story fragment that must be merged to determine the cross-fragment content structure [S9.4].

10 Content structure is merged on an element-by-element basis, merging the last element closed
 11 in the leading story fragment with the first element opened in the trailing story fragment. This
 12 process continues until the closing tag from the leading story fragment no longer matches the
 13 opening tag from the trailing story fragment.

14 <TableCellStructure> elements require special merging, such that all <TableCellStructure>
 15 elements within a <TableRowStructure> element are merged. In order to merge the table cells
 16 and rows correctly, producers MUST specify empty <TableCellStructure> elements for cells that
 17 do not break across story fragments [M9.1].

1 *Example 16-5. Content structure spanning pages*

2 Given the following two StoryFragments parts, consumers can construct the content structure
3 as shown.

```

4     <!-- First StoryFragments part -->
5
6     <StoryFragments
7         xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
8         <StoryFragment FragmentType="Header">
9             <StoryBreak />
10            <ParagraphStructure>
11                <NamedElement NameReference="Block1" />
12            </ParagraphStructure>
13            <StoryBreak />
14        </StoryFragment>
15        <StoryFragment StoryName="Story1" FragmentType="Content">
16            <StoryBreak />
17            <SectionStructure>
18                <TableStructure>
19                    <TableRowGroupStructure>
20                        <TableRowStructure>
21                            <TableCellStructure>
22                                <ParagraphStructure>
23                                    <NamedElement NameReference="Block2" />
24                                    <NamedElement NameReference="Block3" />
25                                </ParagraphStructure>
26                            </TableCellStructure>
27                            <TableCellStructure>
28                                <ParagraphStructure>
29                                    <NamedElement NameReference="Block4" />
30                                </ParagraphStructure>
31                            </TableCellStructure>
32                        </TableRowStructure>
33                        <TableRowStructure>
34                            <TableCellStructure>
35                                <ParagraphStructure>
36                                    <NamedElement NameReference="Block5" />
37                                    <NamedElement NameReference="Block6" />
38                                </ParagraphStructure>
39                            </TableCellStructure>
40                            <TableCellStructure>
41                                <ParagraphStructure>
42                                    <NamedElement NameReference="Block7" />
43                                </ParagraphStructure>
44                            </TableCellStructure>
45                        </TableRowStructure>
46                    </TableRowGroupStructure>
47                </TableStructure>
48            </SectionStructure>
49        </StoryFragment>
50        <StoryFragment FragmentType="Footer">
51            <StoryBreak />
52            <ParagraphStructure>
53                <NamedElement NameReference="Block8" />

```

```

1         </ParagraphStructure>
2         <StoryBreak />
3     </StoryFragment>
4 </StoryFragments>

5
6 <!-- Second StoryFragments part -->
7
8 <StoryFragments
9     xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
10    <StoryFragment FragmentType="Header">
11        <StoryBreak />
12        <ParagraphStructure>
13            <NamedElement NameReference="Block9" />
14        </ParagraphStructure>
15        <StoryBreak />
16    </StoryFragment>
17    <StoryFragment StoryName="Story1" FragmentType="Content">
18        <SectionStructure>
19            <TableStructure>
20                <TableRowGroupStructure>
21                    <TableRowStructure>
22                        <TableCellStructure />
23                        <TableCellStructure>
24                            <ParagraphStructure>
25                                <NamedElement NameReference="Block10" />
26                                <NamedElement NameReference="Block11" />
27                            </ParagraphStructure>
28                        </TableCellStructure>
29                    </TableRowStructure>
30                    <TableRowStructure>
31                        <TableCellStructure>
32                            <ParagraphStructure>
33                                <NamedElement NameReference="Block12" />
34                            </ParagraphStructure>
35                        </TableCellStructure>
36                        <TableCellStructure>
37                            <ParagraphStructure>
38                                <NamedElement NameReference="Block13" />
39                            </ParagraphStructure>
40                        </TableCellStructure>
41                    </TableRowStructure>
42                </TableRowGroupStructure>
43            </TableStructure>
44        </SectionStructure>
45        <StoryBreak />
46    </StoryFragment>
47    <StoryFragment FragmentType="Footer">
48        <StoryBreak />
49        <ParagraphStructure>
50            <NamedElement NameReference="Block14" />
51        </ParagraphStructure>
52        <StoryBreak />
53    </StoryFragment>
54 </StoryFragments>

```

```

1
2   <!-- Resulting merged content structure for Story1 -->
3
4   <SectionStructure>
5     <TableStructure>
6       <TableRowGroupStructure>
7         <TableRowStructure>
8           <TableCellStructure>
9             <ParagraphStructure>
10              <NamedElement NameReference="Block2" />
11              <NamedElement NameReference="Block3" />
12            </ParagraphStructure>
13          </TableCellStructure>
14          <TableCellStructure>
15            <ParagraphStructure>
16              <NamedElement NameReference="Block4" />
17            </ParagraphStructure>
18          </TableCellStructure>
19        </TableRowStructure>
20        <TableRowStructure>
21          <TableCellStructure>
22            <ParagraphStructure>
23              <NamedElement NameReference="Block5" />
24              <NamedElement NameReference="Block6" />
25            </ParagraphStructure>
26          </TableCellStructure>
27          <TableCellStructure>
28            <ParagraphStructure>
29              <NamedElement NameReference="Block7" />
30              <NamedElement NameReference="Block10" />
31              <NamedElement NameReference="Block11" />
32            </ParagraphStructure>
33          </TableCellStructure>
34        </TableRowStructure>
35        <TableRowStructure>
36          <TableCellStructure>
37            <ParagraphStructure>
38              <NamedElement NameReference="Block12" />
39            </ParagraphStructure>
40          </TableCellStructure>
41          <TableCellStructure>
42            <ParagraphStructure>
43              <NamedElement NameReference="Block13" />
44            </ParagraphStructure>
45          </TableCellStructure>
46        </TableRowStructure>
47      </TableRowGroupStructure>
48    </TableStructure>
49  </SectionStructure>

```

50 *end example]*

51 **16.1.2.1 <StoryFragments> Element**

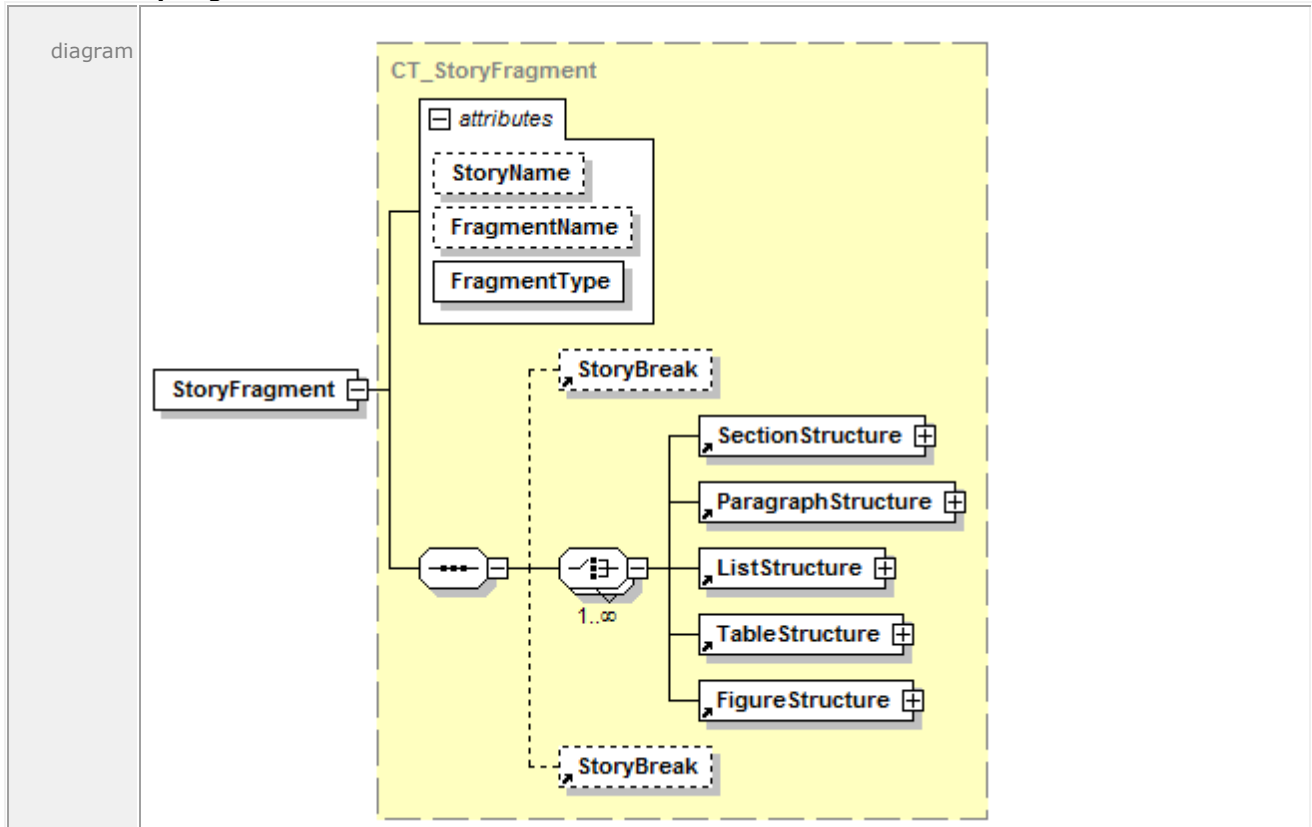
52 element **StoryFragments**

<p>diagram</p>	<pre> classDiagram class StoryFragments class StoryFragment StoryFragments "1" -- "1..∞" StoryFragment </pre>
<p>annotation</p>	<p>The root of a StoryFragments part. Contains all story fragments that appear on a specific page.</p>

- 1 The <StoryFragments> element groups all of the <StoryFragment> elements on a page.

1 **16.1.2.2 <StoryFragment> Element**

2 element **StoryFragment**



attributes	Name	Type	Use	Default	Fixed	Annotation
	StoryName	xs:string	optional			Identifies the story that this story fragment belongs to. If omitted, the story fragment is not associated with any story.
	FragmentName	xs:string	optional			Used to uniquely identify the story fragment.
	FragmentType	<u>ST_FragmentType</u>	required			Specifies the type of content included in the story fragment. Valid values are Content, Header, and Footer.

annotation Specifies the document structural markup that appears on the current page for a single story block.

- 3
- 4 Each <StoryFragment> has a StoryName attribute that associates it with a story defined in the
- 5 DocumentStructure part. It also has a FragmentType attribute, the values for which are Content
- 6 (the default), Header, or Footer.

1 Headers and footers are defined in their own story fragment on each page. These stories do not
 2 specify a StoryName value, so they are essentially unreferenced stories that exist only on a
 3 single page.

4 Producers authoring document structure information SHOULD reference every element of the
 5 fixed page markup that has semantic meaning (such as text or images) in the StoryFragments
 6 parts [S9.5].

7 *Example 16–6. StoryFragments part markup*

8 The following markup describes the StoryFragments part of a one-page document:

```

9     <StoryFragments
10       xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
11       <StoryFragment FragmentType="Header">
12         <StoryBreak />
13         <ParagraphStructure>
14           <NamedElement NameReference="Block13" />
15           <NamedElement NameReference="Block14" />
16         </ParagraphStructure>
17         <StoryBreak />
18       </StoryFragment>
19       <StoryFragment StoryName="Story1" FragmentType="Content">
20         <StoryBreak />
21         <ParagraphStructure>
22           <NamedElement NameReference="Block1" />
23           <NamedElement NameReference="Block2" />
24         </ParagraphStructure>
25         <TableStructure>
26           <TableRowGroupStructure>
27             <TableRowStructure>
28               <TableCellStructure>
29                 <ParagraphStructure>
30                   <NamedElement NameReference="Block3" />
31                   <NamedElement NameReference="Block4" />
32                 </ParagraphStructure>
33               </TableCellStructure>
34               <TableCellStructure>
35                 <ParagraphStructure>
36                   <NamedElement NameReference="Block5" />
37                 </ParagraphStructure>
38               </TableCellStructure>
39             </TableRowStructure>
40           </TableRowGroupStructure>
41         </TableStructure>
42         <SectionStructure>
43           <ParagraphStructure>
44             <NamedElement NameReference="Block6" />
45           </ParagraphStructure>
46           <ParagraphStructure>
47             <NamedElement NameReference="Block7" />
48             <NamedElement NameReference="Block8" />
49           </ParagraphStructure>
50         </SectionStructure>
51         <SectionStructure>
52           <FigureStructure>

```

```

1         <NamedElement NameReference="Block9" />
2     </FigureStructure>
3     <ListStructure>
4         <ListItemStructure>
5             <ParagraphStructure>
6                 <NamedElement NameReference="Block10" />
7             </ParagraphStructure>
8         </ListItemStructure>
9         <ListItemStructure>
10            <ParagraphStructure>
11                <NamedElement NameReference="Block11" />
12            </ParagraphStructure>
13        </ListItemStructure>
14        <ListItemStructure>
15            <ParagraphStructure>
16                <NamedElement NameReference="Block12" />
17            </ParagraphStructure>
18        </ListItemStructure>
19    </ListStructure>
20 </SectionStructure>
21 <StoryBreak />
22 </StoryFragment>
23 <StoryFragment FragmentType="Footer">
24     <StoryBreak />
25     <ParagraphStructure>
26         <NamedElement NameReference="Block15" />
27         <NamedElement NameReference="Block16" />
28         <NamedElement NameReference="Block17" />
29     </ParagraphStructure>
30     <StoryBreak />
31 </StoryFragment>
32 </StoryFragments>

```

33 *end example]*

34 A <StoryFragment> element MAY be identified with a FragmentName attribute to distinguish it
35 from other fragments for the same story on a single page [M2.72].

36 *Example 16–7. Story fragments markup using a fragment name*

```

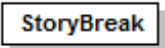
37     <StoryFragments
38         xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure">
39         <StoryFragment
40             StoryName="Story1"
41             FragmentName="Fr1"
42             FragmentType="Content">
43             <StoryBreak />
44             <ParagraphStructure>
45                 <NamedElement NameReference="Block1" />
46                 <NamedElement NameReference="Block2" />
47             </ParagraphStructure>
48             <StoryBreak />
49         </StoryFragment>
50         <StoryFragment
51             StoryName="Story1"
52             FragmentName="Fr2"

```

```
1      FragmentType="Content">
2      <StoryBreak />
3      <ParagraphStructure>
4          <NamedElement NameReference="Block8" />
5      </ParagraphStructure>
6      <StoryBreak />
7  </StoryFragment>
8  </StoryFragments>
9  end example]
```

1 **16.1.2.3 <StoryBreak> Element**

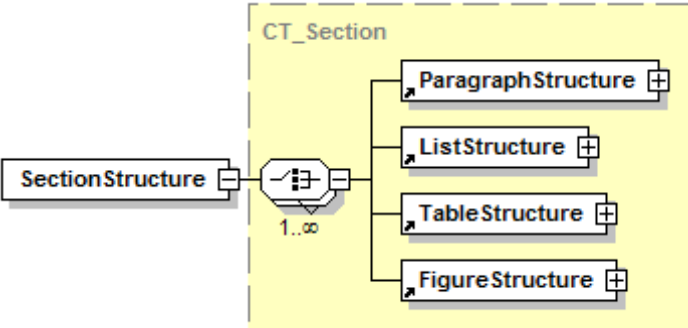
2 element **StoryBreak**

diagram	
annotation	If located at the beginning of a <StoryFragment> definition, indicates that the following markup elements should not be merged with the markup from the previous <StoryFragment>. If located at the end of a <StoryFragment> definition, indicates that the preceding markup elements should not be merged with the subsequent <StoryFragment>.

3 The <StoryBreak> element signals to the consumer not to perform merging across story
4 fragments to determine the content structure.

5 **16.1.2.4 <SectionStructure> Element**

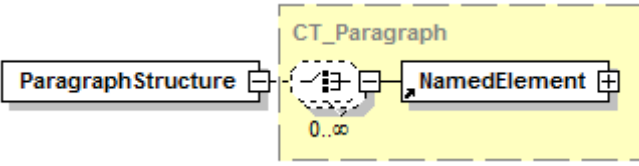
6 element **SectionStructure**

diagram	
annotation	Provides an arbitrary grouping of content structural markup elements.

7 The <SectionStructure> element provides an arbitrary grouping of <Paragraph>,
8 <TableStructure>,<ListStructure>,<FigureStructure> elements.

9 **16.1.2.5 <ParagraphStructure> Element**

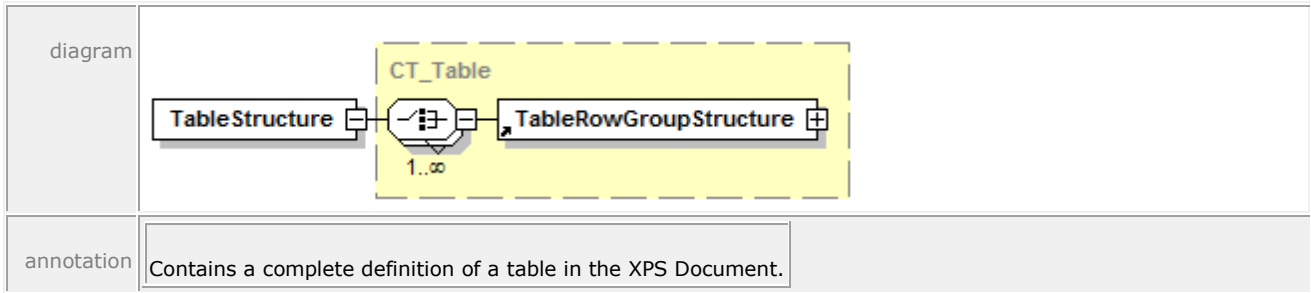
10 element **ParagraphStructure**

diagram	
annotation	Contains the named elements that constitute a single paragraph.

11 A <ParagraphStructure> element describes the list of <NamedElement> elements that
12 constitute a single paragraph.

1 **16.1.2.6 <TableStructure> Element**

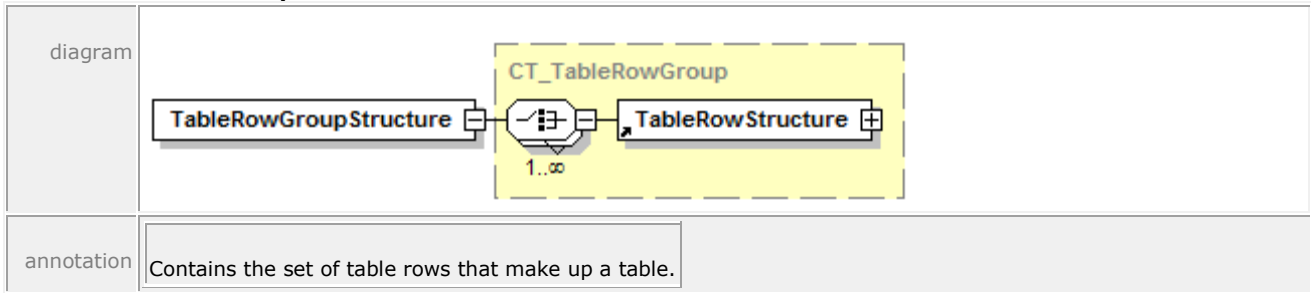
2 element **TableStructure**



3 A <TableStructure> element is the complete definition of a table. An implementation MAY use it
 4 to build special functionality, such as row or column selection [O9.5].

5 **16.1.2.7 <TableRowGroupStructure> Element**

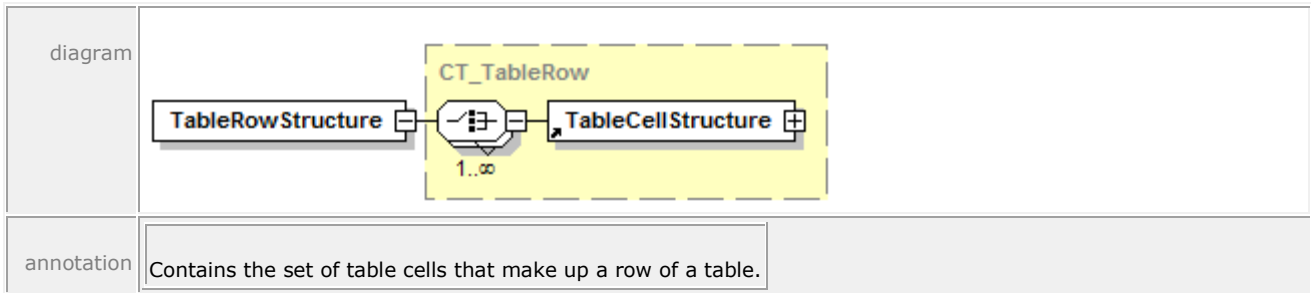
6 element **TableRowGroupStructure**



7 A <TableRowGroupStructure> element is REQUIRED in order to specify a set of
 8 <TableRowStructure> elements [M2.72].

9 **16.1.2.8 <TableRowStructure> Element**

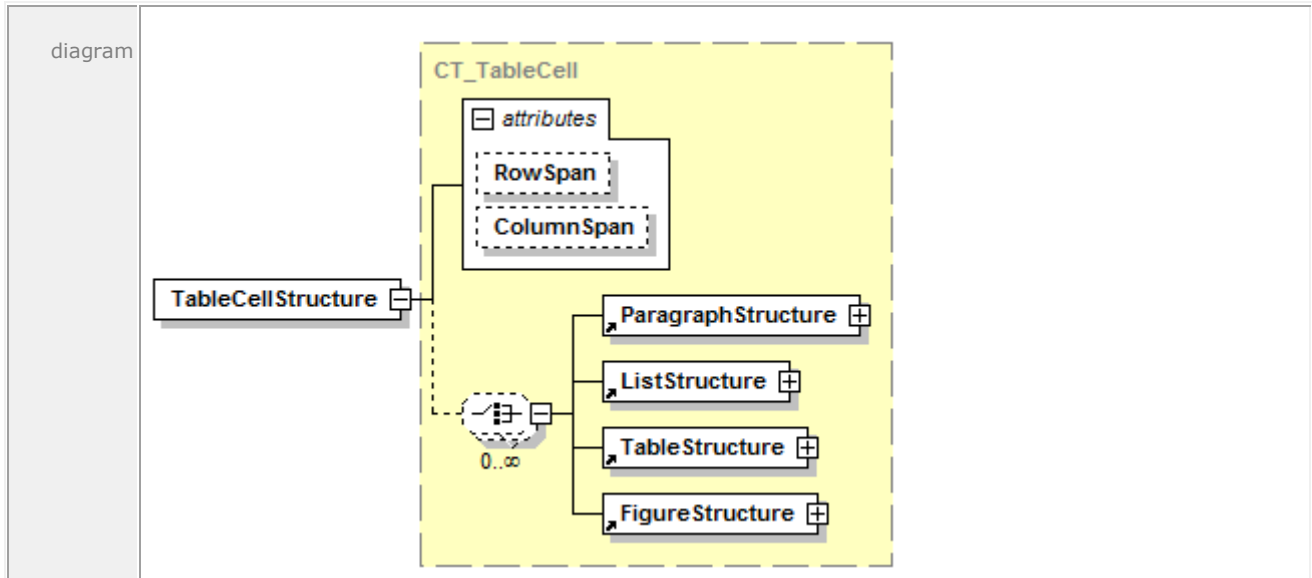
10 element **TableRowStructure**



11 This element groups <TableCellStructure> child elements that define a single row of a table.

1 **16.1.2.9 <TableCellStructure> Element**

2 element **TableCellStructure**



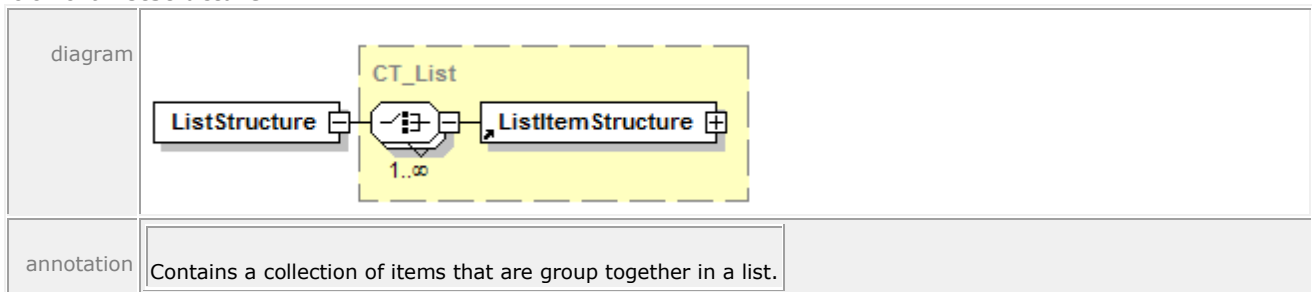
attributes	Name	Type	Use	Default	Fixed	Annotation
	RowSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of rows this cell spans, or merges into a single cell.
	ColumnSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of columns this cell spans, or merges into a single cell.

annotation: Contains the elements that occupy a single cell of a table.

3 This element defines the appearance of a table cell. It MAY contain nested `<TableStructure>`
 4 elements [M2.72].

5 **16.1.2.10 <ListStructure> Element**

6 element **ListStructure**

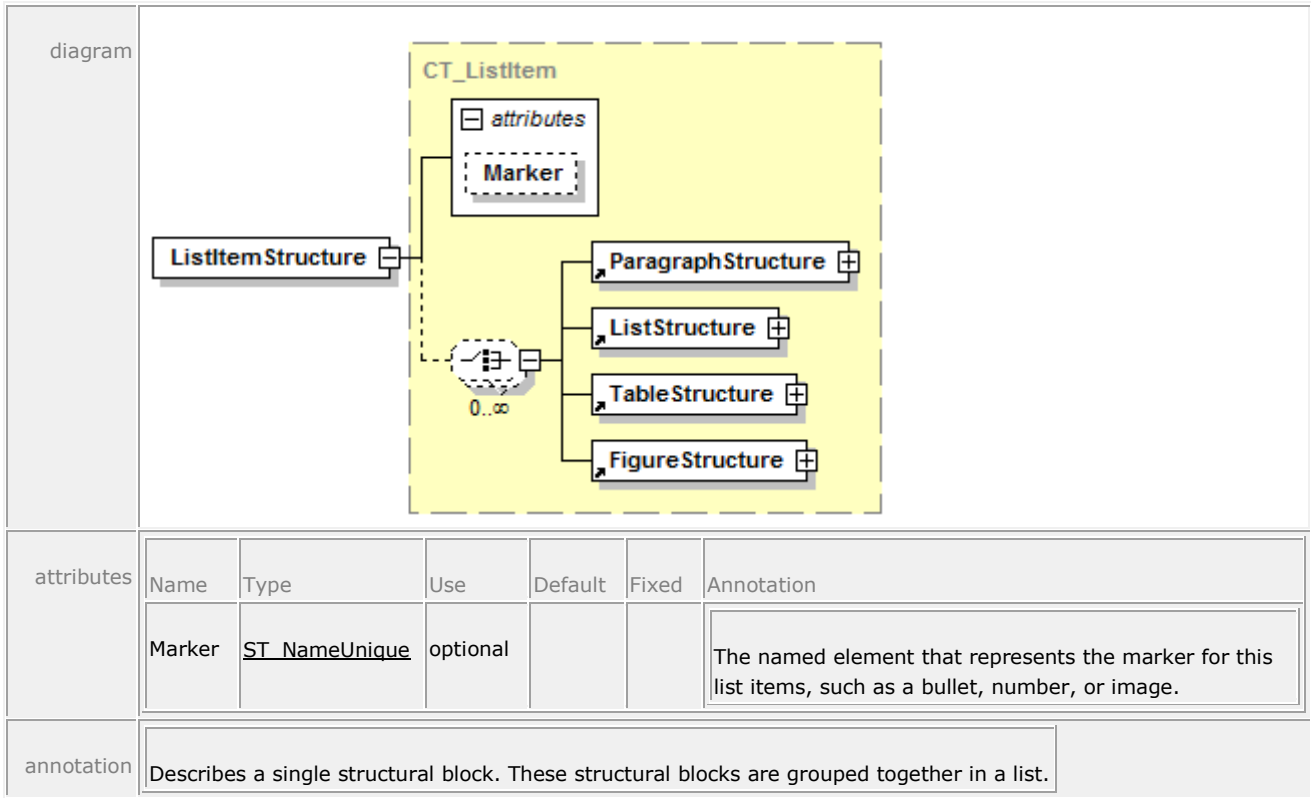


annotation: Contains a collection of items that are group together in a list.

7 The `<ListStructure>` element is the complete definition of a list of related items.

1 **16.1.2.11 <ListItemStructure> Element**

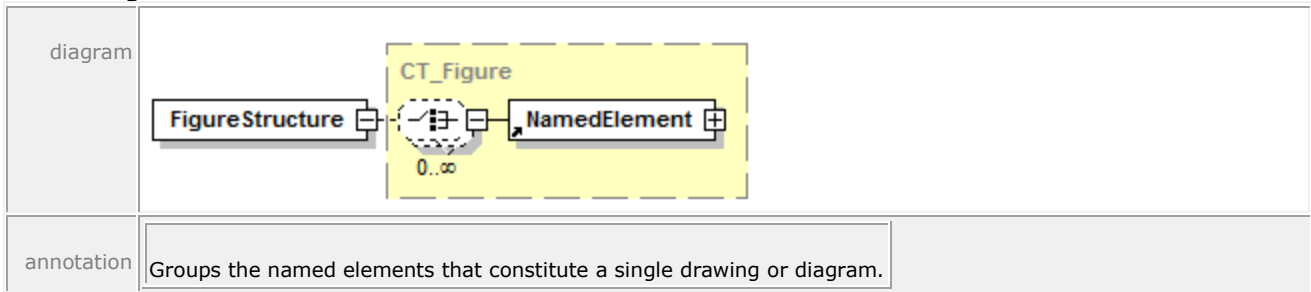
2 element **ListItemStructure**



3 A <ListItemStructure> element defines a single item in a list.

4 **16.1.2.12 <FigureStructure> Element**

5 element **FigureStructure**



6 A <FigureStructure> element includes a group of named elements that comprise a single drawing or diagram.

7

1 **16.1.2.13 <NamedElement> Element**2 element **NamedElement**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	NameReference	<u>ST_Name</u>	required			Identifies the named element in the FixedPage part markup that is referenced by the document structure markup.
annotation	All document structure is related to the fixed page markup using this element. The <NamedElement> points to a single markup element contained in the fixed page markup.					

3 A <NamedElement> references a specific element in the fixed page by using the NameReference
4 attribute to specify an element in the fixed page markup with a corresponding name.

5 If the targeted fixed page uses markup compatibility markup that changes the presence of
6 certain named elements, the StoryFragments part should also use it in order to reference each
7 element in either representation.

8 **16.2 Hyperlinks**

9 If consumers enable user interactivity, they SHOULD support hyperlink activation and
10 addressing [S9.6].

11 **16.2.1 Hyperlink Activation**

12 Hyperlinks are specified inline on any <Canvas>, <Path>, or <Glyphs> element by means of
13 the FixedPage.NavigateUri attribute. The value of the attribute is the destination URI. If
14 hyperlinked <Path> or <Glyphs> elements are rendered as overlapping on the page,
15 consumers MUST treat the topmost element as the only hyperlink that can be activated in the
16 overlapping region [M9.2].

17 When activating a hyperlink, consumers SHOULD load the specified resource if they understand
18 the URI type. If the URI is an internal reference to the XPS Document, consumers SHOULD
19 navigate to the URI [S9.7].

20 If a producer specifies a FixedPage.NavigateUri attribute on a <Canvas> element, consumers
21 MUST treat all child elements of that canvas as having an associated hyperlink [M9.3]. Child or
22 descendant elements can override this value with their own FixedPage.NavigateUri attribute.

23 Relative internal hyperlinks between FixedPage parts MUST specify, at a minimum, the named
24 address relative to the FixedDocument part [M9.4].

1 Producers can mark any <FixedPage>, <Canvas>, <Path>, or <Glyphs> element as an
2 addressable location within the XPS Document by specifying a value for the Name attribute. The
3 name SHOULD be unique within the scope of the fixed document [S9.8]. If it is not unique, only
4 the first occurrence of the named address is addressable.

5 These elements, if specified as a <VisualBrush.Visual> property element are not addressable by
6 a hyperlink.

7 It is RECOMMENDED that Name attribute values be unique within an entire fixed document
8 sequence [S9.9]. If they are not, only the first occurrence of the named address is addressable
9 from an external location. Internal hyperlinks can specify a named element fragment relative to
10 a particular fixed document, but consumers MAY interpret such a URI relative to the entire fixed
11 document sequence instead [O9.6].

12 In order to be addressable by either a hyperlink or the document outline, the named address
13 MUST appear in the <PageContent.LinkTargets> element in the fixed document [M9.5]. If a
14 named address appears in the <PageContent.LinkTargets> element in the fixed document but
15 is not found in the Name attribute of an element within the associated fixed page, consumers
16 MUST treat the top of the associated fixed page as the named address [M9.6]. If the named
17 address in a URI fragment is not found, consumers MUST ignore the fragment portion of the
18 URI [M9.7].

19 *Example 16–8. A relative, internal, named-address hyperlink*

```
20 FixedPage.NavigateUri=" ../MyDocument.fdoc#MyAddress"
```

21 *end example]*

22 16.2.2 Hyperlink Addressing

23 XPS Documents specify two forms of URI fragment identifiers to address locations within an
24 XPS Document. The first is a named address. [*Example*: "http://xps/MyPackage#MyAddress",
25 where "http://xps/MyPackage" is an XPS Document and "MyAddress" is a named address within
26 the document. *end example*] The second is an absolute page number within the XPS Document.
27 [*Example*: "http://xps/MyPackage#15", where "15" references the FixedPage part associated
28 with the fifteenth <PageContent> entry among all the fixed documents in the fixed document
29 sequence. *end example*]

30 Page number fragment identifiers refer to the absolute page number (1-based) in the fixed
31 document sequence. [*Example*: If an XPS Document has a 3-page fixed document, followed by
32 a 10-page fixed document, followed by an 8-page fixed document, the fragment identifier
33 "#15" refers to the second page of the third fixed document in the fixed document sequence.
34 *end example*] Internal references MUST specify a page address relative to the fixed document
35 sequence [M9.8].

36 *Example 16–9. A relative internal page address hyperlink*

```
37 FixedPage.NavigateUri=" ../ ../ ../MyDocSeq.fdoc#12"
```

38 *end example]*

39 16.2.3 Name Attribute

40 The Name attribute contains a string value that identifies the current element as a named,
41 addressable point for the purpose of hyperlinking. The Name attribute is optional. Names
42 SHOULD be unique within a fixed document [S9.8], and it is RECOMMENDED that they be

1 unique within a fixed document sequence [S9.9]. The Name attribute MUST NOT be specified on
2 any children of a <ResourceDictionary> element [M9.10].

3 If the Name attribute is specified, producers SHOULD also create a corresponding <LinkTarget>
4 element in the FixedDocument part within the <PageContent> element that links to the parent
5 fixed page [S9.10]. Consumers MAY ignore this attribute [O9.7], but devices that support user
6 interaction with the contents of XPS Documents SHOULD support hyperlinks [S9.6].

7 The Name value, if specified, MUST meet the following requirements [M2.72]:

- 8 1. The initial character MUST be an underscore character or a letter, that is, it falls within
9 the Lu, Ll, Lo, Lt, and Nl categories [M2.72].
- 10 2. Trailing characters MUST be an underscore character or a letter or number, that is, they
11 fall within the Lu, Ll, Lo, Lt, Nl, Mn, Mc, and Nd categories [M2.72].

12 [*Note*: These requirements match those of XML identifiers with additional restrictions. *end note*]

13 The category abbreviations, as defined within the Unicode Character Database, are partially
14 reproduced in Table 16–2.

15 *Table 16–2. Unicode character categories*

Abbreviation	Description
Lu	Letter, uppercase
Ll	Letter, lowercase
Lt	Letter, titlecase
Lo	Letter, other
Mn	Mark, non-spacing
Mc	Mark, spacing combining
Nd	Number, decimal
Nl	Number, letter

16 **16.2.4 FixedPage.NavigateUri Attribute**

17 The FixedPage.NavigateUri attribute associates a hyperlink URI with an element, making it a
18 hyperlink source. Its value can be a relative or absolute URI that addresses a resource that is
19 internal or external to the XPS Document package, respectively. The base URI used to resolve a
20 relative URI is that of the FixedPage part in which the element with the FixedPage.NavigateUri
21 attribute appears. Therefore, a hyperlink to a destination within the fixed document of the
22 source MUST specify the destination in the context of the FixedDocument part [M9.4].
23 [*Example*: “./FixedDoc1.fdoc#MyDestination”. *end example*] A destination in the same fixed
24 document SHOULD be expressed as a relative URI [S9.11].

25 The FixedPage.NavigateUri attribute is OPTIONAL [M2.72]. It SHOULD be included *only* if the
26 element is intended to be a hyperlink. Consumers MAY ignore this attribute [O9.8], but devices
27 that support user interaction with the contents of XPS Documents SHOULD support hyperlinks
28 [S9.6].

16.3 Selection

Viewing consumers that support interactivity MAY support selection and copying [O9.9]. If selection is supported, consumers SHOULD provide a visual cue over or around selected elements [S9.12]. Selection order within an XPS Document SHOULD follow reading order [S9.13].

Consumers MAY use the FragmentType attribute of the <StoryFragment> element to determine selection behavior, such as disallowing selection of both the page header and the page contents while allowing independent selection within those stories [O9.10].

16.4 Accessibility

Accessibility refers to features that are important to provide equal access to XPS Documents for users of all abilities. One common example of an accessibility application is a screen reader, which reads the contents of a document aloud for vision-impaired individuals.

16.4.1 Reading Order

In the absence of document structure information provided in the XPS Document, consumers MAY infer the reading order from the position of elements on the page [O9.11], but SHOULD, at minimum, rely on the markup order to determine reading order [S9.14]. Producers SHOULD order the markup in FixedPage parts to reflect the order in which it is intended to be read [S9.15]. When document structure information is present, consumers SHOULD rely on the order of appearance of named elements in the content structure markup to determine reading order [S9.16].

The RECOMMENDED reading order of a page-centric application is as follows [S9.17]:

- Order the content by page.
- Within a page, order by story fragment in the order the <StoryFragment> elements are specified in the StoryFragments part for that page. Producers SHOULD order <StoryFragment> elements in their intended reading order [S9.18].
- Within a <StoryFragment> element, order by <NamedElement> reference.
- Append all un-referenced elements that appear in the fixed page markup, ordered by markup order.

Although producers SHOULD reference every element of the fixed page markup in the content structure markup [S9.10], consumers MUST expose every element of the fixed page markup to an accessibility interface in the determined reading order, even if the elements are not referenced in the content structure markup [M9.9].

Consumers MAY use the FragmentType attribute of the <StoryFragment> element to determine reading order by interpreting elements that have FragmentType values of Header and Footer as belonging first or last in the reading order, respectively [O9.12].

The RECOMMENDED reading order of a story-centric application is as follows [S9.19]:

- Order content by story in the sequence the <Story> elements appear in the DocumentStructure part. Producers SHOULD order <Story> elements in their intended reading order [S9.20].

- 1 • Within a story, order <StoryFragmentReference> elements in the sequence they appear
2 in the DocumentStructure part. Producers SHOULD order <StoryFragmentReference>
3 elements in their intended reading order [S9.21].
- 4 • Within a story fragment, order by <NamedElement> references in the StoryFragments
5 part markup.
- 6 • Append all un-referenced elements that appear in the fixed page markup, ordered by
7 page number, then markup order.

8 **16.4.2 Screen Reader Applications**

9 Screen reader applications read the contents of the document aloud. A screen reader consumer
10 SHOULD read the document according to its reading order [S9.22]. The application SHOULD
11 use the UnicodeString attribute of each <Glyphs> element [S9.23]. In addition, screen readers
12 MAY inspect the Indices attribute to resolve potential ambiguities [O9.13].

13 If the screen reader provides features to navigate the document by structural elements, such as
14 paragraphs or table rows, it SHOULD use any document structure information included in the
15 XPS Document [S9.24].

16 If the screen reader provides features to describe images, it SHOULD read the text provided in
17 the AutomationProperties.Name and AutomationProperties.HelpText attributes [S9.25].

18 If the screen reader provides features to describe hyperlink addresses, it SHOULD read the text
19 provided in the FixedPage.NavigateUri attribute [S9.26].

20 **16.4.3 Text Alternatives for Graphics and Images**

21 Images and graphics SHOULD specify text alternatives for images and graphics to make this
22 content accessible to vision-impaired individuals [S9.27]. There are short and long textual
23 descriptions, specified in the AutomationProperties.Name and AutomationProperties.HelpText
24 attributes of <Path> and <Canvas>, respectively.

25 The AutomationProperties.Name attribute SHOULD contain a short description of the basic
26 contents of the image or vector graphic [S9.27]. [*Example: "A sitting dog." end example*]The
27 AutomationProperties.HelpText attribute can contain a more detailed description of the image or
28 graphic. [*Example: "A cocker spaniel with brown eyes, golden fur, and its tongue hanging out.
29 It is sitting on a beanbag directly facing the camera." end example*]

30 An image SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText
31 attributes on the <Path> element that is filled with an <ImageBrush> [S9.28]. These attributes
32 describe the content specified by the ImageSource attribute of the <ImageBrush> element.

33 A vector graphic (a collection of one or more <Path> elements representing a single drawing)
34 SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText attributes only
35 once, directly on a <Canvas> element wrapping the <Path> elements comprising the graphic
36 [S9.29].

37 Individual <Path> elements that do not provide any semantic meaning (such as a line between
38 sections or outlining a table) SHOULD NOT specify these text alternative attributes [S9.27].

39

17. XPS Document Package Features

The XPS Document format extends package-level interleaving and digital signatures as described in the OPC specification.

17.1 Interleaving Optimizations

Interleaving concerns the physical organization of XPS Documents, rather than their logical structure. It allows consumers to linearly process the bytes that make up a physical package from start to finish, without regard for context. In other words, consumers can make correct determinations about the types of logical parts and the presence of relationships on a logical part when consuming packages in a linear fashion. Consumers are never required to return to previously encountered parts and revise their determination of the content type or presence of relationships.

Interleaving is OPTIONAL [O10.1]. However, if the XPS Document is interleaved, these rules SHOULD be followed:

- The Content Types stream SHOULD be interleaved according to the recommendations in the OPC specification [S10.1].
- PrintTicket parts SHOULD be written to the package before the part to which they are attached [S10.2].
- The portion of the relationship data attaching the PrintTicket to a part SHOULD be written to the package before the part to which it is attached or in close proximity to the part to which it is attached [S10.3].
- If no PrintTicket settings are specified for a FixedDocumentSequence, FixedDocument, or FixedPage part, an empty PrintTicket part SHOULD be attached to the part, and the portion of the relationship data attaching the empty PrintTicket SHOULD be written to the package before the part to which it is attached or in close proximity to the part to which it is attached [S10.4].
- The last piece of the Relationships part for a FixedPage part SHOULD be written to the package in close proximity to the first piece of the FixedPage part [S10.5].

Following these recommendations allows more efficient processing by certain consumers. Not following these recommendations could result in less efficient processing by most consumers because they will need to wait until all parts required to process a part (attached PrintTicket, required resources) have been consumed. However, consumers MUST be prepared to correctly process packages in which the PrintTicket or the portion of the relationship data attaching the PrintTicket appears in the package after the affected part [M10.1].

Consumers can choose to parse an XPS Document in a head-first or tail-first manner. Tail-first parsing reveals certain package errors earlier, such as inconsistencies between the ZIP central directory and local file headers. Head-first XPS Document consumers SHOULD attempt to detect inconsistent packages as soon as possible and SHOULD generate an error message, even if they have already processed the pages that resulted in the error [S10.18]. Head-first consumers that discard parts would need to retain the name and length of any discarded part to comply with this recommendation.

1 | [\[Note: Streaming and handling of discard control are complicated significantly by any](#)
2 | [requirement for out-of-order page handling, such as in the production of booklets. *end note*\]](#)

1 **17.1.1 Empty PrintTicket**

2 An empty PrintTicket has the following form:

```
3 <psf:PrintTicket
4 xmlns:psf="http://schemas.microsoft.com/windows/2003/08/printing/printschemafra
5 work" version="1"/>
```

6 It is RECOMMENDED that one empty PrintTicket be shared for all parts that attach an empty
7 PrintTicket [S10.6].

8 **17.1.2 Optimizing Interleaving Order**

9 Producers MAY optimize the interleaving order of parts to help consumers avoid stalls during
10 read-time streaming, and to allow consumers to manage their memory resources more
11 efficiently [O10.2].

12 The optimization strategy is suggested by the consumer architecture. Therefore, interleaving
13 optimization is typically implemented by a software component such as a driver or filter that is
14 specific to (or aware of) the consumer architecture.

15 **17.1.2.1 Single-Threaded Parsing Architectures**

16 An optimal interleaving scheme for consumers with a single-threaded parsing model interleaves
17 parts so that each part that is required to consume a single page (FixedPage, images, and
18 fonts) is contained in the package in its entirety, prior to the FixedPage part being referenced
19 from the FixedDocument part's markup.

20 Single-threaded parsing architectures typically require more run-time memory resources than
21 multi-threaded parsing architectures because the context in which a resource is used is
22 unknown at the time the resource is received. This requires deferred processing and additional
23 buffering.

24 *[Note: When interleaving entities containing XML markup, such as the DiscardControl part, the*
25 *Content Types stream, and the FixedDocument part, there is no guarantee that XML element*
26 *boundaries will align with piece boundaries in the physical package. This adds a complexity to*
27 *single-threaded parsing architectures: the parser must be pre-emptable. Certain existing XML*
28 *parser implementations might require a pre-tokenization step. end note]*

29 *Example 17-1. Optimized interleaving for a single-threaded parsing architecture*

30 The following markup describes a sequence of two fixed documents, the first having two
31 FixedPage parts and the second having one FixedPage part:

32

Part/Piece	Markup
Font1.ttf	...binary font data...
Other resources	...resource data...
Page1	<FixedPage xmlns="http://schemas.microsoft.com /xps/2005/06" ...> <Glyphs FontURI="Font1.ttf"/> </FixedPage>
Page1.rels	<Relationships xmlns=

	<pre>"http://schemas.openxmlformats.org/package/2006/re lationships"> <Relationship Type= "http://schemas.microsoft.com/xps/2005/06 /required-resource" Target="Font1.ttf"/> </Relationships></pre>
FixedDocument1/[0].piece	<pre><FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06"> <PageContent Source="Page1"/></pre>
Sequence1/[0].piece	<pre><FixedDocumentSequence xmlns= "http://schemas.microsoft.com/xps/2005/06"> <DocumentReference Source="FixedDocument1"/></pre>
_rels/.rels/[0].piece	<pre><Relationships xmlns= "http://schemas.openxmlformats.org/package/2006/re lationships"> <Relationship Type="StartPart" Target="Sequence1"/></pre>
Page2	<pre><FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06" ...>...</FixedPage></pre>
FixedDocument1/[1].last.piece	<pre><PageContent Source="Page2"/> </FixedDocument></pre>
Page3	<pre><FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06" ...>...</FixedPage></pre>
FixedDocument2	<pre><FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06"> <PageContent Source="Page3"/> </FixedDocument></pre>
Sequence1/[1].last.piece	<pre><DocumentReference Source="FixedDocument2" /> </FixedDocumentSequence></pre>
_rels/.rels/[1].last.piece	<pre></Relationships></pre>

1 *end example]*

2 **17.1.2.2 Multi-Threaded Parsing Architectures**

3 An optimal interleaving scheme for consumers with a multi-threaded parsing model interleaves
4 parts so that each resource part that is required to consume a single page (images and fonts) is
5 contained in the package after the FixedPage part referencing it.

6 Multi-threaded parsing architectures typically require less run-time memory resources than
7 single-threaded parsing architectures because the context in which resources appear is fully
8 determined and, therefore, resources can be processed immediately.

9 [*Note: When interleaving entities containing XML markup, such as the DiscardControl part, the*
10 *content type stream, and the FixedDocument part, there is no guarantee that XML element*
11 *boundaries will align with piece boundaries in the physical package. A multi-threaded parsing*
12 *architecture is naturally suited to address this problem. end note]*

1 *Example 17-2. Optimized interleaving for a multi-threaded parsing architecture*

2 The following markup describes a sequence of two FixedDocument parts, the first having two
3 FixedPage parts and the second having one FixedPage part:

4

Part/Piece	Markup
_rels/.rels/[0].piece	<pre><Relationships xmlns="http://schemas.microsoft.com/package /2005/06/relationships"> <Relationship Type="StartPart" Target= "Sequence1"/></pre>
Sequence1/[0].piece	<pre><FixedDocumentSequence xmlns= "http://schemas.microsoft.com/xps/2005/06"> <DocumentReference Source="FixedDocument1"/></pre>
FixedDocument1/[0].piece	<pre><FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06"> <PageContent Source="Page1"/></pre>
Page1.rels	<pre><Relationships xmlns= "http://schemas.openxmlformats.org/package/2006/re lationships"> <Relationship Type= "http://schemas.microsoft.com/xps/2005/06 /required-resource" Target="Font1.ttf"/> </Relationships></pre>
Page1	<pre><FixedPage xmlns="http://schemas.microsoft.com /xps/2005/06" ...> <Glyphs FontURI="Font1.ttf"/> </FixedPage></pre>
Font1.ttf	...binary font data...
Other resources	...resource data...
FixedDocument1/[1].last.pie ce	<pre><PageContent Source="Page2"/></pre>
Page2	<pre><FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06" ...>...</FixedPage></pre>
FixedDocument1/[2].last.pie ce	<pre></FixedDocument></pre>
Sequence1/[1].last.piece	<pre><DocumentReference Source="FixedDocument2" /> </FixedDocumentSequence></pre>
FixedDocument2	<pre><FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06"> <PageContent Source="Page3"/> </FixedDocument></pre>
Page3	<pre><FixedPage xmlns="http://schemas.microsoft.com/xps/2005/06"</pre>

```
...>...</FixedPage>
```

```
_rels/.rels/[1].last.piece </Relationships>
```

1 *end example]*

2 **17.1.3 Consuming Interleaved Packages**

3 Consumers MUST be able to consume packages regardless of their interleaving structure
 4 [M10.2]. Consumers that lack the resources to process a part MUST indicate an error condition
 5 [M10.3]. Such a resource constraint exists when a consumer lacks sufficient memory resources
 6 to hold enough of the package to resolve all the references required to process a part.

7 To address resource constraints:

- 8 • Consumers MAY discard FixedPage parts once they have been processed [O10.3]
- 9 • Consumers MAY discard FixedDocument and FixedDocumentSequence parts after all their
 10 child elements and their closing tags have been processed [O10.4].
- 11 • In the absence of explicit directives to the contrary (see §17.1.4), consumers MAY
 12 discard parts as directed by the DiscardControl part [O10.5]. Consumers MUST NOT
 13 discard any other parts [*Example*: Such as parts containing fonts, images, or other
 14 resources *end example*] unless they have the ability to access the parts again [M10.4].

15 If a consumer encounters a reference to an unknown part, it MUST continue to receive further
 16 bytes of the package until the unknown part has been transmitted *or* until the end of the
 17 package is reached (indicating an error condition) [M10.5].

18 **17.1.4 Consumers with Resource Constraints**

19 To produce an XPS Document for streaming consumption by consumers with limited memory
 20 resources, some producers MAY choose a suitable interleaving order by modeling the resource
 21 management behavior of the consumer [O10.6]. These producers, referred to as *drivers*, must
 22 have specific knowledge of the XPS Document consumer. Due to resource constraints, some
 23 consumers are unable to consume arbitrary XPS Documents and always require assistance from
 24 an external driver.

25 When some consumers with limited memory resources receive a XPS Document in a streaming
 26 fashion, there might be an opportunity to discard parts when necessary and reload them again
 27 when needed. Producers, such as drivers, that target such consumers SHOULD follow these
 28 steps [S10.7]:

- 29 • Conservatively model the memory usage of the device.
- 30 • Interleave pieces of parts in the correct order.
- 31 • Decide when certain parts can be discarded by the consumer and inform the consumer
 32 within the package stream (see §17.1.4.1).
- 33 • Add to the package a uniquely named copy of a resource that could have been discarded,
 34 if the resource is referenced by a part sent later in the stream. Those later references
 35 are also updated to refer to the new copy of the resource.

36 **17.1.4.1 DiscardControl Part**

37 In addition to optimally ordering interleaved parts, producers can support consumers with
 38 resource constraints by means of the DiscardControl part. The DiscardControl part is a well-
 39 known part containing a list of resources that are safe for the consumer to discard.

1 DiscardControl parts are stored in XPS Documents in an interleaved fashion, allowing a
 2 resource-constrained consumer to discard a part as soon as it appears in the DiscardControl
 3 part. DiscardControl parts are targeted with a DiscardControl package relationship, as specified
 4 in §H. There MUST NOT be more than one DiscardControl package relationship [M10.23]. The
 5 DiscardControl part MUST NOT reference itself [M10.6]; doing so is considered an error.

6 DiscardControl parts that are not well-formed SHOULD NOT be processed and an error SHOULD
 7 NOT be reported [S10.8]. The consumer MAY decide to ignore the malformed DiscardControl
 8 part in its entirety or from the first malformed node onward [O10.7].

9 In some cases, producers might rewrite the contents of a package so that parts are provided
 10 more than once, allowing consumers to discard a part in order to free resources for additional
 11 processing. Each instance of a part MUST be stored as a new, uniquely named part in the
 12 package [M10.24].

13 *Example 17-3. A DiscardControl part*

```

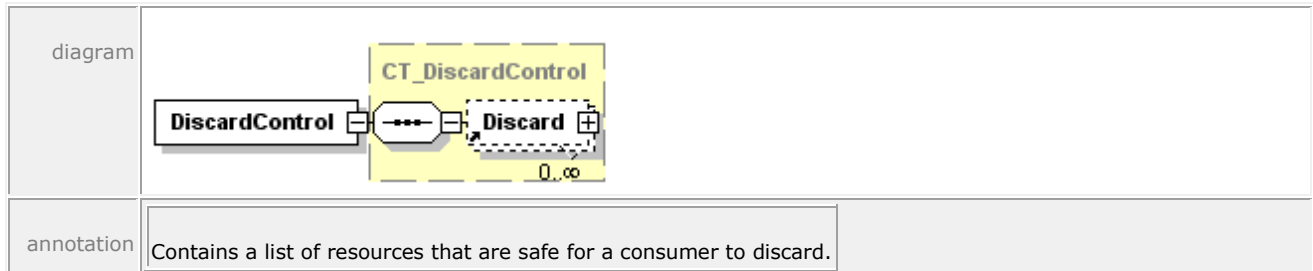
14 <DiscardControl xmlns="http://schemas.microsoft.com/xps/2005/06/discard-
15 control">
16   <!-- May discard partname1 as soon as starting to process
17     page11.xaml -->
18   <Discard SentinelPage="/page11.xaml" Target="/partname1" />
19   <!-- May discard partname2 as soon as starting to process
20     page13.xaml -->
21   <Discard SentinelPage="/page13.xaml" Target="/partname2" />
22   ...
23 </DiscardControl>

```

24 *end example]*

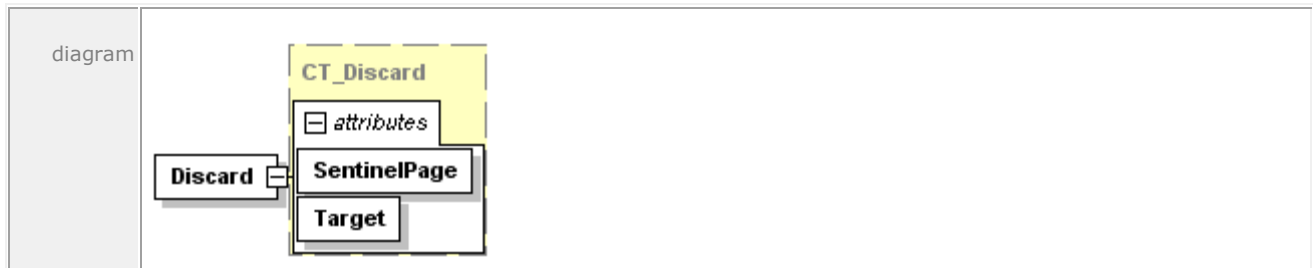
25 **17.1.4.1.1 <DiscardControl> Element**

26 element **DiscardControl**



27 **17.1.4.1.2 <Discard> Element**

28 element **Discard**



attributes	Name	Type	Use	Default	Fixed	Annotation
	SentinelPage	xs:anyURI	required			
Target	xs:anyURI	required				The resource that can be safely discarded.
annotation	Identifies a resource that can be safely discarded by a resource-constrained consumer.					

1 Parts that can be discarded are identified in a <Discard> element by the Target attribute value,
 2 which is expressed as relative to the package root, and by the SentinelPage attribute value,
 3 which identifies the first FixedPage part that no longer requires the discarded part. (The
 4 processing order for FixedPage parts is implied by the order of <PageContent> element
 5 references in the FixedDocument part. Therefore, the value of the SentinelPage attribute is
 6 unambiguous.)

7 If either the Target attribute or the SentinelPage attribute contain an invalid reference (refer
 8 outside the package), the respective <Discard> element MUST be ignored [M10.7]. If a
 9 <Discard> element is encountered where either or both of the Target attribute and SentinelPage
 10 attribute identify a part which has not been processed yet (is still unknown), the <Discard>
 11 element SHOULD be retained until both parts identified by the Target attribute and SentinelPage
 12 attribute have been processed or until the end of the package is reached [S10.9].

13 17.1.5 Interleaving Optimizations and Digital Signatures

14 In general, it is not feasible to produce well-ordered, interleaved ZIP packages *and* apply digital
 15 signatures in a way that enables reasonable consumption scenarios for the following reasons:

- 16 • The digital signature parts must be known to consumers before they process other signed
 17 parts because the selected hash-methods and transforms must be known. A streaming
 18 consumer might not be able to access part data after it has been processed for printing.
- 19 • Producers cannot create the digital signature parts before producing the signed
 20 packages.
- 21 • There are cyclic dependencies with signed relationship parts containing the relationship
 22 to the signature parts themselves.

23 Therefore, when adding a digital signature to an interleaved package, producers of digitally
 24 signed documents that are intended for streaming consumption SHOULD add all digital
 25 signature parts and the package relationship to the digital signature parts at the beginning of
 26 the package, before adding any other part [S10.10].

27 17.2 Digital Signatures

28 The digital signature specification for XPS Documents is described in the OPC specification. It
 29 allows users to sign arbitrary parts, relationship parts, and individual relationships. Although
 30 XPS Documents also use these digital signature mechanisms, they have a specific signature
 31 policy and a specific signing mechanism for documents containing co-signing requests.

17.2.1 Signature Policy

This specification defines the signature policy that governs the methods of signing and verifying signatures for XPS Documents. The XPS Document signature policy includes a specific set of signing rules and validity rules. All producers and consumers signing and verifying signatures for end users or applications MUST adhere to these rules consistently [M10.8] to ensure that end users can rely on applications to display accurate signature information.

When signing a document, users can choose to make any of the following actions invalidate the signature:

- Editing core properties
- Adding signatures

Consumers MUST NOT prevent an end user from taking an action solely because doing so will invalidate an existing signature [M10.9]. Consumers SHOULD, however, inform the end user if an action they are going to take will invalidate an existing signature [S10.11].

17.2.1.1 Signing Rules

An XPS Document MUST be considered signed according to the XPS Document signing policy, regardless of the validity of that signature, if the following *signing rules* are followed [M10.10]:

1. The following parts MUST be signed [M10.10]:
 - a. The <SignedInfo> portion of the Digital Signature XML Signature part containing this signature.
 - b. The FixedDocumentSequence part that is the target of the Start Part package relationship.
 - c. All FixedDocument parts referenced in the markup of the FixedDocumentSequence part. (Adding a FixedDocument part to a signed XPS Document will invalidate the signature.)
 - d. All FixedPage parts referenced by all signed FixedDocument parts.
 - e. All parts associated with each signed FixedPage part by means of a Required Resource relationship (such as fonts, images, color profiles, remote resource dictionaries).
 - f. All DocumentStructure parts associated via a Document Structure relationship with all signed FixedDocument parts.
 - g. All StoryFragments parts associated via Story Fragments relationship with all signed FixedPage parts.
 - h. All SignatureDefinitions parts associated via a Signature Definitions relationship with any signed FixedDocument part. (Once a document is signed, adding any new signature definitions will invalidate the signature.)
 - i. All Thumbnail parts associated via a Thumbnail relationship from the package root or with any signed FixedPage or FixedDocument part.
2. The following parts MAY be signed [M10.10]:
 - a. The CoreProperties part.
 - b. The Digital Signature Origin part.
 - c. A Digital Signature Certificate part.
 - d. PrintTicket parts.

- 1 e. DiscardControl parts.
- 2 3. All relationships with the following RelationshipTypes (see §H) MUST be signed [M10.10]:
- 3 a. StartPart relationship from the package root
- 4 b. DocumentStructure relationship from a FixedDocument part
- 5 c. StoryFragments relationship from a FixedPage part
- 6 d. Digital Signature Definitions relationship from a FixedDocument part
- 7 e. Required Resource relationship from a FixedPage part
- 8 f. Restricted Font relationship from a FixedDocument part
- 9 g. Thumbnail relationship from a FixedPage part, a FixedDocument part, or the package
- 10 root
- 11 4. All relationships with the following RelationshipTypes MUST be signed if their Target part
- 12 is signed [M10.10]:
- 13 a. Core Properties relationship
- 14 b. Digital Signature Origin relationship
- 15 c. Digital Signature Certificate relationship from a Digital Signature XML Signature part
- 16 d. PrintTicket relationship
- 17 e. DiscardControl relationship
- 18 5. Relationships with the following RelationshipTypes MAY be signed as a group (they MUST
- 19 NOT be signed individually) [M10.10]:
- 20 a. All Digital Signature XML Signature relationships from the Digital Signature Origin part
- 21 (signing all relationships of this RelationshipType will cause this signature to break
- 22 when a new signature is added).
- 23 6. All of the above-referenced parts and relationships MUST be signed using a single digital
- 24 signature [M10.10].

25 An XPS Document MUST NOT be considered signed according to the XPS Document signing
26 policy if [M10.11]:

- 27 1. Any part not covered by the signing rules above is included in the signature.
- 28 2. Any relationship not covered by the signing rules above is included in the signature.

29 An XPS Document digital signer MUST NOT sign an XPS Document that contains content (parts
30 or relationships parts) to be signed that defines the Markup Compatibility namespace ~~but~~when
31 the signer does not fully understand all elements, attributes, and alternate content
32 representations introduced through the markup compatibility mechanisms [M10.12]. An XPS
33 Document digital signer MAY choose not to sign any content (parts or relationships parts) that
34 defines the Markup Compatibility namespace, even ~~if~~when the content is fully understood
35 [O10.8].

36 17.2.1.2 Signing Validity

37 An XPS Document digital signature MUST be shown as an *incompliant digital signature* if
38 [M10.13]:

- 39 • It violates any of the signing rules described above regarding parts or relationships that
- 40 MUST or MUST NOT be signed.

- 1 An XPS Document digital signature MUST be shown as a *broken digital signature* if [M10.14]:
- 2 • It is not an incompliant digital signature, but the signature fails the signature validation
- 3 routines described in the OPC.
- 4 An XPS Document digital signature MUST be shown as a *questionable digital signature* if any of
- 5 the following are true [M10.15]:
- 6 • It is not an incompliant or broken digital signature, but the certificate cannot be
- 7 authenticated against the certificate authority.
- 8 • It is not an incompliant or broken digital signature, but the signed content (parts and
- 9 relationships) contain elements or attributes from an unknown namespace introduced
- 10 through the Markup Compatibility mechanisms.
- 11 An XPS Document digital signature MAY be shown as a questionable digital signature if [O10.9]:
- 12 • It is not an incompliant or broken digital signature, but contains some other detectable
- 13 problem at the discretion of the consumer.
- 14 An XPS Document digital signature MUST be shown as a *valid digital signature* if [M10.16]:
- 15 • It is not an incompliant, broken, or questionable digital signature.

16 **17.2.1.3 Adding Signatures**

17 XPS Documents MAY be signed more than once [O10.10]. A user who signs an XPS Document

18 might or might not want to allow any additional signing of the document. To prohibit additional

19 signatures in an XPS Document, the signing application MUST sign all the Digital Signature

20 Origin part's relationships of relationship type Digital Signature with the same signature as the

21 rest of the content [M10.17].

22 **17.2.1.4 Certificate Store**

23 XPS Document signatures MUST NOT refer to a remote certificate store (certificate not

24 contained in the XPS Document). All certificates MUST be stored in the XPS Document either as

25 a Certificate part or in the Digital Signature XML Signature part [M10.18].

26 **17.2.1.5 Printing Signed Documents**

27 When printing signed documents, the PrintTicket setting JobDigitalSignatureProcessing SHOULD

28 be used to control the digital signature processing behavior [S10.12]. Producers MAY include

29 the JobDigitalSignatureProcessing setting in the job-level PrintTicket within the XPS Document

30 content [O10.11]. Consumers SHOULD process this PrintTicket setting, if present [S10.12]. For

31 more information, see the Print Schema specification.

32 *Table 17-1. JobDigitalSignatureProcessing PrintTicket settings*

Name	Description
PrintInvalidSignatures	Print the job regardless of the validity of the digital signatures. Digital signatures can be ignored.
PrintInvalidSignaturesWithErrorReport	Print the job regardless of the validity of the digital signatures. In the event an invalid signature is encountered, an error page should print at the end of the job. Digital signatures cannot be ignored.
PrintOnlyValidSignatures	Print the job only if all digital signatures are valid. Digital signatures cannot be ignored.

1 17.2.2 Signature Definitions

2 In some workflow scenarios, documents must be signed as a means of approving their content.
3 [*Example*: Document producers might be required to sign their documents in order to provide
4 proof of authenticity. *end example*] In other cases, reviewers might be required to co-sign
5 content before it can be submitted for publication. These requirements can be fulfilled with a
6 digitally signed XPS Document.

7 Whereas the XPS package model supports the signing of arbitrary content in a package, an XPS
8 Document signing workflow requires additional features, including the ability to specify co-
9 signature requirements and to include workflow-specific signature information in the document.
10 XPS Document authors and signing parties provide such information in an XML *signature*
11 *definition*.

12 Signature definitions are represented by <SignatureDefinition> elements within a single
13 <SignatureDefinitions> element.

1 *Example 17-4. A SignatureDefinitions part*

```

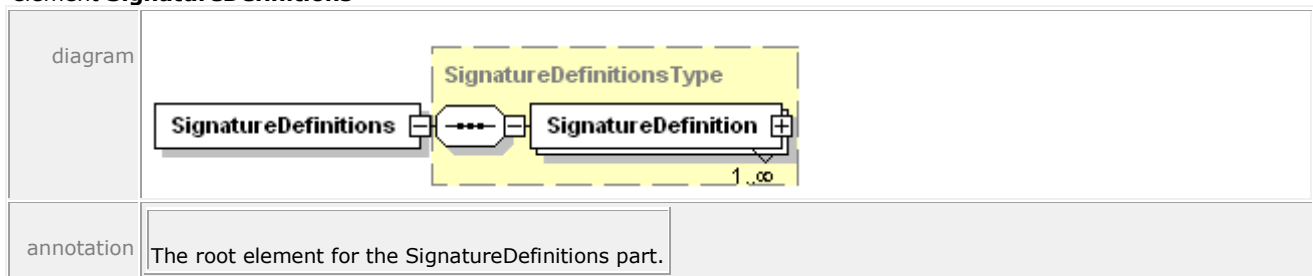
2   <SignatureDefinitions xmlns="http://schemas.microsoft.com/xps/2005/06/
3     signature-definitions">
4     <SignatureDefinition SignerName="Dorena Paschke"
5       SpotID="0e0a7abb-48c9-595d-77db-305e84a05fc3">
6       <SpotLocation
7         PageURI="/Documents/1/Pages/2.fpage"
8         StartX="0.0"
9         StartY="0.0" />
10      <Intent>I have read and agree</Intent>
11      <SignBy>2005-08-20T23:59:59Z</SignBy>
12      <SigningLocation>Redmond, WA</SigningLocation>
13    </SignatureDefinition>
14  </SignatureDefinitions>

```

15 *end example]*

16 17.2.2.1 <SignatureDefinitions> Element

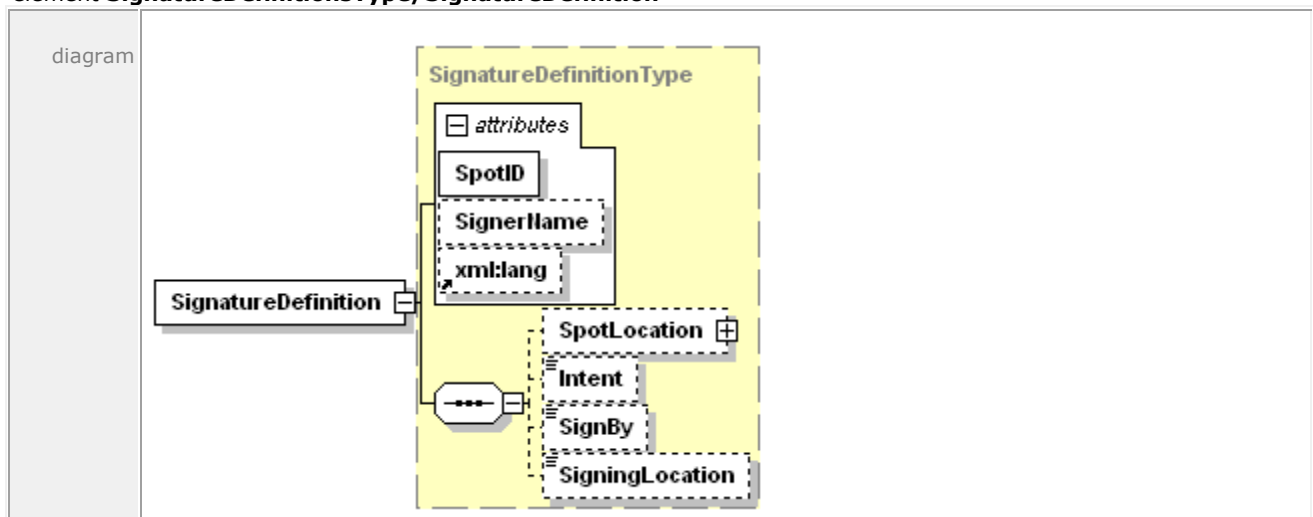
17 element **SignatureDefinitions**



18 If the SignatureDefinitions part exists, it MUST contain only one <SignatureDefinitions>
 19 element [M2.72]. The XML namespace for the <SignatureDefinitions> element is specified
 20 in §H.1.

21 17.2.2.2 <SignatureDefinition> Element

22 element **SignatureDefinitionsType/SignatureDefinition**



attributes	Name	Type	Use	Default	Fixed	Annotation
	SpotID	xs:ID	required			A globally unique identifier for this signature spot.
	SignerName	xs:string				A string representing the identity of the individual who is requested to sign the XPS Document, or the name of the individual who has signed the XPS Document.
	xml:lang					Specifies the language used for the current element and its descendants. The language is specified according to RFC 3066.
annotation	A single signature definition.					

1 If the SignatureDefinitions part exists, there MUST be *at least* one <SignatureDefinition>
 2 element [M2.72].

3 **17.2.2.2.1 SpotID Attribute**

4 The SpotID attribute is REQUIRED [M2.72]. This attribute MAY be used to link an existing
 5 signature to the <SignatureDefinition> element [O10.12]. The value of this attribute MUST be
 6 globally unique to ensure that a Signature part can be linked to only one <SignatureDefinition>
 7 element [M2.72]. To link a <SignatureDefinition> to a signature, the value of the SpotID MUST
 8 be specified in the Id attribute of the corresponding <Signature> element in the Digital
 9 Signature XML Signature part [M10.19]. For more information, see "Digital Signatures" in the
 10 OPC specification.

11 **17.2.2.3 <SpotLocation> Element**

12 element **SignatureDefinitionType/SpotLocation**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	PageURI	xs:anyURI	required			Specifies the page on which the signature spot should be displayed.
	StartX	xs:double	required			Specifies the x coordinate of the origin point (upper-left corner) on the page where the signature spot should be displayed.

	StartY	xs:double	required			Specifies the y coordinate of the origin point (upper-left corner) on the page where the signature spot should be displayed.
annotation	Specifies where a consumer should place a signature spot.					

1 The <SpotLocation> element is OPTIONAL [M2.72]. It specifies where an XPS Document viewer
 2 should place a visual representation or *signature spot* to indicate that a digital signature has
 3 been applied or requested. The viewing consumer SHOULD use the values specified in this
 4 element. Due to space and rendering limitations, producers MUST NOT assume that consumers
 5 will use these values [M10.20]. If the location specified by this element is not used, it is
 6 RECOMMENDED that consumers choose a location that does not contain any page content
 7 [S10.13].

8 The size and shape of the signature spot are determined by the consumer. Consumers MAY
 9 choose a size and shape based on the desired display information and page content [O10.13].
 10 However, it is RECOMMENDED that they render signature spots as consistently sized rectangles
 11 that include the signer name, the intent, the signing location, and the scope of the XPS
 12 Document to be signed [S10.14]. It is also RECOMMENDED that the signature spot be a
 13 clickable area used to launch the digital signing process [S10.15].

14 *Figure 17-1. A sample signature spot*



15

16 17.2.2.4 <Intent> Element

17 element **SignatureDefinitionType/Intent**

diagram	
annotation	A string that represents the intent to which the signing party agrees when signing the document.

18 Consumers MUST display the full value of the <Intent> element to the signing party, either in
 19 the signature spot or through some other mechanism [M10.21].

20 [Note: Consumers that wish to display signature spots must consider the implications of
 21 supporting any Unicode character that can be specified in the <Intent> element. They must
 22 also make decisions about the appropriate font face and size to use as well as determine the
 23 proper layout and interactivity of the signature spot. These decisions are implementation-
 24 specific. *end note*]

1 **17.2.2.5 <SignBy> Element**2 element **SignatureDefinitionType/SignBy**

diagram	
annotation	The date and time by which the requested party is to sign the XPS Document.

3 If specified, the consumer SHOULD NOT allow the signing party to sign the document using this
 4 particular signature spot after the date and time specified [S10.16]. The date and time MUST
 5 be specified in UTC time, using the format "Complete date plus hours, minutes and seconds"
 6 described in the W3C Note "Date and Time Formats" [M10.22], [*Example*: "2006-12-
 7 31T23:59:59Z" for 11:59 PM (UTC) on December 31, 2006. *end example*]

8 **17.2.2.6 <SigningLocation> Element**9 element **SignatureDefinitionType/SigningLocation**

diagram	
annotation	The legal location where the document is signed.

10 The <SigningLocation> element MAY be set by the original producer of the XPS Document or by
 11 the signing party at the time of signing [O10.14].

12 **17.3 Core Properties**

13 XPS Documents use the Core Properties part described in the OPC. The core properties
 14 specified in that part SHOULD refer to the entire fixed payload, including the root
 15 FixedDocumentSequence part and the compilation of all FixedDocument parts it references
 16 [S10.17].

18. Rendering Rules

The set of rules described here ensures precise and consistent rendering of XPS Document markup across various implementations. Producers MUST generate XPS Documents that can be accurately rendered by following the rules described in this clause [M11.1]. Consumers MUST adhere to the rules described in this clause when rendering XPS Documents [M11.1]. In addition to rules for visual elements, implementation limits are also discussed.

18.1 Coordinate System and Rendering Placement

In the x,y coordinate system, one unit is initially equal to 1/96", expressed as a real number. The initial origin of the coordinate system is the top left corner of the fixed page. The x -coordinate value increases from left to right; the y -coordinate value increases from top to bottom.

A RenderTransform property can be specified on any path, glyphs, or canvas to apply an affine transform to the current coordinate system.

A Transform property can be specified on any visual brush, image brush, linear gradient brush, radial gradient brush, or path geometry to apply an affine transform to the current coordinate system.

18.1.1 Page Dimensions

The logical page dimensions correspond to the media size specified in the application page layout and are specified by the Width and Height attributes of the <FixedPage> element. Further optional attributes on the <FixedPage> element are used to specify details about the areas of the fixed page that contain rendered content. For more information, see §10.3.

The physical page dimensions correspond to the media size specified for printing. The physical page dimensions are specified in the PrintTicket PageMediaSize keyword.

The interaction of the logical page dimensions and the physical page dimensions is print-specific. For more information, see §10.3.3 and §10.3.4.

18.1.2 Rounding of Coordinates

All computations on coordinate values SHOULD be performed with at least single floating-point precision [S11.1]. Final conversion (after all transforms have been computed) to device coordinates SHOULD retain at least as much fractional precision as a 28.4 fixed-point representation before performing pixel coverage calculations [S11.1].

Very high resolution devices MAY use lower fractional precision to represent device coordinates [O11.1].

When converting from real-number coordinate values to device coordinate values, rounding is performed according to the following rule:

$$\text{coord}_d = \text{ROUND}(\text{coord}_r * 16.0) / 16$$

Where coord_r expresses a real-number coordinate value and coord_d expresses a device coordinate value.

1 **18.1.3 Transforms**

2 XPS Document markup supports affine transforms as expressed through the `RenderTransform`
 3 and `Transform` properties. An affine transform is represented as a list of six real numbers: `m11`,
 4 `m12`, `m21`, `m22`, `dx`, `dy`. (For markup details, see §14.4.)

5 The full matrix is as follows:

$$\begin{bmatrix} m11 & m12 & 0 \\ m21 & m22 & 0 \\ dx & dy & 1 \end{bmatrix}$$

6 A given x,y coordinate is transformed with a render transform to yield the resulting coordinate
 7 x',y' by applying the following computations:

$$\begin{aligned} 8 \quad x' &= x * m11 + y * m21 + dx \\ 9 \quad y' &= x * m12 + y * m22 + dy \end{aligned}$$

10 When rendering a child or descendant element, the effective transform used for rendering is the
 11 concatenation of all the transforms specified by the `RenderTransform` or `Transform` property on
 12 parent or ancestor elements, starting from the outermost ancestor.

13 Non-invertible effective transforms can be specified in markup or occur as a result of limited
 14 numerical precision during concatenation. If a non-invertible transform is encountered during
 15 rendering, consumers MUST omit rendering the affected element and all of its child and
 16 descendant elements [M11.2].

17 If a non-invertible transform is encountered on a brush (as specified directly on the brush, as a
 18 result of the `Viewbox` or `Viewport` attributes, or through concatenation), the brush is treated
 19 according to §18.7.1.

20 The `Width` and `Height` values specified in the `Viewbox` and `Viewport` attributes of an
 21 `<ImageBrush>` or `<VisualBrush>` element MUST NOT be negative [M2.72].

22 If a non-invertible transform is encountered on a geometry (as specified directly on the
 23 geometry or through concatenation), the geometry MUST be considered to contain no area
 24 [M11.3].

25 A final, device-dependent step using the horizontal resolution and vertical resolution of the
 26 device converts the resulting coordinates x',y' to device coordinates x'',y'' , as follows:

$$\begin{aligned} 27 \quad x'' &= x' * R_x/96 \\ 28 \quad y'' &= y' * R_y/96 \end{aligned}$$

29 Where R_x is the horizontal resolution and R_y is the vertical resolution of the device, specified in
 30 device pixels per inch.

31 **18.1.4 Pixel Center Location, Pixel Placement, and Pixel Inclusion**

32 A pixel covers the range from x to $x+1$.

33 An *ideal* consumer implementation SHOULD render pixels in an 8x8 sub-pixel space, perform an
 34 8x8 box filter sampling, and set the pixel to the resulting color value [S11.2]. Other
 35 implementations MAY use different rendering logic as long as it closely approximates this logic
 36 [O11.2].

1 When rendering a shape, a *practical* implementation (such as a bi-tonal printing device)
2 SHOULD turn on each pixel whose center (at $x+0.5$) is covered by the shape, or is touched by
3 the shape with the shape extending beyond the pixel center in the positive x or y direction of
4 the device [S11.3]. Devices MAY use sub-pixel masking instead [O11.3].

5 By definition, a shape with an area width of 0 (that is, no included area) does not touch or
6 cover any pixel centers. A stroke with a width of 0 is treated in the same manner.

7 As a result of these rules, the behavior for very thin lines is implementation-specific:

- 8 • An implementation capable of anti-aliasing MAY draw a thin line in a way that blends with
9 the background to varying degrees [O11.4].
- 10 • A bi-tonal implementation on a printer MAY draw thin lines, or apply half-toning,
11 depending on the desired output quality [O11.5]. If such an implementation chooses to
12 draw thin lines, then it MAY choose to draw them with drop outs, following requirement
13 S11.3 in §18.1.4 above, or as solid rules of 1 pixel thickness [O11.26].

14 [Note: Also see §18.6.12 for discussion of thin strokes. *end note*]

15 **18.1.5 Maximum Placement Error**

16 When rendering geometries, consumers SHOULD render curves so they appear smooth from a
17 normal viewing distance [S11.4]. Producers MUST NOT assume a specific placement error for
18 curve decomposition or rely on side-effects of a specific consumer implementation [M11.4].

19 **18.1.6 Pixel Placement for Glyphs**

20 Regardless of other rules expressed here, consumers MAY apply pixel placement rules
21 optimized for character rendering to individual glyphs in a <Glyphs> element [O11.6]. Such
22 rules can result from font hinting applied by the typeface scaler used by a consumer
23 implementation.

24 **18.1.7 Abutment of Shapes**

25 When no anti-aliasing is used, abutting shapes that share the same device coordinates for the
26 end-points and control-points of an edge SHOULD be rendered without overlap and without
27 gaps [S11.5]. Ideally, an implementation SHOULD also follow this rule for shapes that are
28 mathematically abutting without sharing device coordinates for end-points and control-points of
29 edges [S11.5].

30 **18.1.8 Clipping Behavior**

31 Clipping occurs as if a mask were created from the clip geometry according to the pixel
32 placement rules defined in §18.1.4. An ideal consumer SHOULD create such a mask in an 8x8
33 sub-pixel space and subsequently draw only those sub-pixels of a shape that correspond to
34 "ON" sub-pixels in the mask [S11.6].

35 A practical implementation (such as a bi-tonal printing device) SHOULD create a pixel mask
36 according to Point2 of §18.1.4, and subsequently draw only those pixels of a shape that
37 correspond to "ON" pixels in the mask. In creating the mask and drawing the shape, the
38 abutment of shapes rule SHOULD be observed so that no pixel of the shape is drawn that would
39 not have been drawn if the clip geometry were another abutting shape [S11.7]. Devices MAY
40 use sub-pixel masking instead [O11.3].

18.2 Implementation Limits

XPS Document markup does not assume fixed implementation limits. However, consumers can have specific implementation limits imposed by their operating environment. XPS Document markup has been designed so that even complex pages can be represented accurately and with high fidelity.

A typical consumer implementation SHOULD be able to process markup with the characteristics indicated in Table 18–1 [S11.8]. Encountering markup with characteristics outside of the consumer-specific implementation limits MUST cause an error condition [M11.5].

Table 18–1 provides the RECOMMENDED minimum requirements for individual elements [S11.8]. Consumers also have limits on the total number of elements, as imposed by available memory. Producers SHOULD produce only XPS Documents that stay within these implementation limits [S11.8].

In order to process pages that contain a large number of elements, consumers MAY implement support for the DiscardControl part in order to discard elements that have already been processed [O10.5].

Table 18–1. Recommended minimum processing requirements

Characteristic	Type	Limit	Description
Coordinates/transformation matrix elements	Real number	+/- 10 ¹²	Largest and smallest coordinate values. Calculations involving numbers close to this limit within a few orders of magnitude are likely to be inaccurate.
Smallest representable non-zero value		+/- 10 ⁻¹²	Coordinate values closest to 0 without rounding to 0. Calculations involving numbers close to this limit within a few orders of magnitude are likely to be inaccurate.
Required precision for coordinates		Single floating point	Coordinates are real numbers and SHOULD be computed with at least single floating point precision [S11.1].
Nested <Canvas> elements		16	Depth of nested <Canvas> elements.
Nested <VisualBrush> elements		16	Depth of nested <VisualBrush> elements within the Visual property. If the nesting level is higher than the limit, a consumer SHOULD attempt to flatten the nested content to a bitmap representation rather than failing to draw [S11.10].
Total number of points in a path figure		100,000	
Total number of points in a segment		100,000	
Total number of points in a geometry		100,000	
Total number of elements per page		1,000,000	
Number of glyphs per		5,000	

Characteristic	Type	Limit	Description
<hr/>			
<Glyphs> element			
Number of elements in a single resource dictionary		10,000	
Total number of elements in all resource dictionaries of an individual page		10,000	
Total number of resource dictionaries in nested canvas scope		Number of nested <Canvas> elements + 1	The <FixedPage> element and each nested <Canvas> element can have at most one associated <ResourceDictionary> element
Number of gradient stops in a gradient brush		100	
Number of fixed documents in a fixed document sequence		1,000	
Number of fixed pages in a fixed document		1,000,000	
Number of dash-gap segments in StrokeDashArray property		No preset limit	Practical number of dash-gap segments depends on the StrokeThickness and the total length of the stroked path.
Total size of XPS Document markup per page	Bytes	64,000,000	Total size of markup after removing all unnecessary whitespace (according to the schema in B), assuming markup elements are specified in the default namespace without namespace prefixes, and assuming the most compact representation of all attributes using abbreviated syntax where possible.

1 18.3 Gradient Computations

2 To ensure the greatest possible consistency among consumers, gradients SHOULD be rendered
3 according to the guidelines described in this subclause [S11.11].

4 18.3.1 All Gradients

5 Linear gradients and radial gradients share a common set of recommended operations for pre-
6 processing gradient stops and blending colors. These are described below.

7 18.3.1.1 Gradient Stop Pre-Processing

8 Consumers SHOULD pre-process gradient stops for all gradients using the following steps
9 [S11.12]:

- 10 1. Sort all gradient stops by their respective offset values in ascending order. When two or
11 more gradient stops have the same offset value, preserve their relative order from the
12 markup while sorting. When more than two gradient stops have the same offset value,
13 remove all but the first and last gradient stops having the same offset value.
- 14 2. If no gradient stop with an offset of 0.0 exists,

- 1 a. And no gradient stop with an offset less than 0.0 exists, create an artificial gradient
2 stop having an offset of 0.0 and a color of the gradient stop with the smallest offset
3 value.
- 4 b. And a gradient stop with an offset less than 0.0 exists and a gradient stop with an
5 offset greater than 0.0 exists, create an artificial gradient stop having an offset of 0.0
6 and a color interpolated between the two gradient stops surrounding 0.0. Discard all
7 gradient stops with an offset less than 0.0.
- 8 c. And a gradient stop with an offset less than 0.0 exists and no gradient stop with an
9 offset greater than 0.0 exists, create an artificial gradient stop having an offset of 0.0
10 and a color of the gradient stop with the largest offset value. Discard all gradient stop
11 elements with an offset less than 0.0.
- 12 3. If no gradient stop with an offset of 1.0 exists,
 - 13 a. And no gradient stop with an offset of greater than 1.0 exists, create an artificial
14 gradient stop having an offset of 1.0 and a color equal to the color of the gradient stop
15 with the largest offset value.
 - 16 b. And a gradient stop with an offset greater than 1.0 exists and a gradient stop with an
17 offset less than 1.0 exists, create an artificial gradient stop having an offset of 1.0 and
18 a color interpolated between the two surrounding gradient stops. Discard all gradient
19 stops with an offset greater than 1.0.
 - 20 c. And a gradient stop with an offset greater than 1.0 exists and no gradient stop with an
21 offset less than 1.0 exists, create an artificial gradient stop having an offset of 1.0 and
22 a color equal to that of the gradient stop with the smallest offset value. Discard all
23 gradient stops with an offset greater than 1.0.

24 18.3.1.2 Blending Colors

25 If any gradient stops use an sRGB or scRGB color specification or the consumer does not
26 understand the PageBlendColorSpace PrintTicket setting, consumers SHOULD blend colors
27 between gradient stops in the color space indicated by the ColorInterpolationMode attribute of the
28 gradient brush [S11.13]. If none of the gradient stop elements uses an sRGB or scRGB color
29 specification and the consumer understands the PageBlendColorSpace PrintTicket setting, the
30 PageBlendColorSpace PrintTicket setting SHOULD be used [S11.13].

31 The function used for blending is:

32 `BLEND(offset, clo, chi)`

33 Where the offset is between 0 and 1. c_{lo} and c_{hi} designate the color values for an offset of 0
34 and 1, respectively.

35 If a ColorInterpolationMode value of SRgbLinearInterpolation is used, the BLEND() function
36 SHOULD convert the color values to sRGB first, and then perform a linear interpolation between
37 them [S11.14].

38 If a ColorInterpolationMode value of ScRgbLinearInterpolation is used, the BLEND() function
39 SHOULD convert the color values to scRGB first, and then perform a linear interpolation
40 between them [S11.15].

41 If blending is performed in the color space identified by the PageBlendColorSpace PrintTicket
42 setting, it SHOULD be a linear, channel-by-channel blend operation [S11.13].

1 In the presence of transformations or when individual gradient stops are very close (separated
 2 by a few pixels or less in the device space), the local color gradient at the offset used in the
 3 BLEND() function might be large, resulting in a large change over the extent of a single device
 4 pixel. In this case, it is RECOMMENDED that the BLEND() function interpolate the gradient over
 5 the extent of each device pixel [S11.16]. However, the behavior MAY differ from this
 6 recommendation in an implementation-specific manner [O11.7] and, therefore, producers
 7 SHOULD NOT rely on a specific effect for such dense gradient specifications [S11.16].

8 As a consequence of this interpolation, radial gradients that define the gradient origin on or
 9 outside ellipse create an "outside" area that can be rendered inconsistently. The radial
 10 gradients that are affected are those that define multiple gradient stops that are of different
 11 colors and are very close in Offset value to 0.0 or 1.0 (the gradient end points), for radial
 12 gradients with a SpreadMethod value of Repeat or Reflect, respectively. For these affected
 13 gradients, consumers MAY use an interpolated color value for the outside area [O11.8].
 14 Depending on the resolution, this can result in different colors than those defined by the
 15 gradient end points. The closer a gradient stop is to the affected gradient end point, the more
 16 the rendering results on different consumers and at different display resolutions can differ.
 17 Producers SHOULD therefore either avoid such close gradient stops to the gradient end point
 18 when specifying radial gradients where the outside area is visible or avoid specifying radial
 19 gradients with a gradient origin on or outside the ellipse (in which case there is no outside area)
 20 to ensure consistent rendering results [S11.17].

21 18.3.2 Linear Gradients

22 Consumers SHOULD render an element filled with a linear gradient brush such that the
 23 appearance is the same as if the following steps had been taken [S11.11]:

- 24 1. Transform the StartPoint and EndPoint attribute values using the current effective render
 25 transform (including the render transform for the element being filled by the linear
 26 gradient brush and the brush's transform itself).
- 27 2. If the SpreadMethod value is Pad, the colors of points on the line defined by the StartPoint
 28 and EndPoint attributes are defined by interpolating the coordinates linearly, and each
 29 color component (such as R, G, B for sRGB and scRGB) as well as the alpha component
 30 is interpolated between the component values of the closest enclosing gradient stops:

```
31 For each offset (real number)  $t < 0$ :
32 {
33    $x(t) = (EndPoint_x - StartPoint_x) * t + StartPoint_x$ 
34    $y(t) = (EndPoint_y - StartPoint_y) * t + StartPoint_y$ 
35    $c(t) = c_{first}$ 
36    $a(t) = a_{first}$ 
37 }
```

38 Where c is the color component and a is the alpha component. c_{first} are the color
 39 component values of the first gradient stop (after sorting) and a_{first} is the alpha
 40 component value at the first gradient stop (after sorting).

```
41 For each offset (real number)  $0 \leq t \leq 1$ :
42 {
43    $x(t) = (EndPoint_x - StartPoint_x) * t + StartPoint_x$ 
44    $y(t) = (EndPoint_y - StartPoint_y) * t + StartPoint_y$ 
45    $c(t) = BLEND((t - t_{10}) / (t_{hi} - t_{10}), c_{10}, c_{hi})$ 
46    $a(t) = [(t - t_{10}) / (t_{hi} - t_{10})] * (a_{hi} - a_{10}) + a_{10}$ 
47 }
```

1 Where t_{lo} and t_{hi} are the offsets, c_{lo} and c_{hi} are the color component values at the
 2 closest enclosing gradient stops (that is, $t_{lo} \leq t \leq t_{hi}$) and a_{lo} and a_{hi} are the alpha
 3 component values at the closest enclosing gradient stops ($t_{lo} \leq t \leq t_{hi}$).

4 For each offset (real number) $t > 1$:

```
5 {
6   x(t) = (EndPointx-StartPointx)*t+StartPointx
7   y(t) = (EndPointy-StartPointy)*t+StartPointy
8   c(t) = clast
9   a(t) = alast
10 }
```

11 Where c_{last} are the color component values of the last gradient stop (after sorting) and
 12 a_{last} is the alpha component value at the last gradient stop (after sorting).

- 13 3. If the SpreadMethod value is Repeat, the colors of points on the line defined by the
 14 StartPoint and EndPoint attributes are defined by interpolating the coordinates linearly, and
 15 each color component (such as R, G, B for sRGB and scRGB) as well as the alpha
 16 component is interpolated between the component values of the closest enclosing
 17 gradient stops:

18 For each repetition (all integers) N :

```
19 {
20   For each offset (real number)  $0 \leq t < 1$ :
21   {
22     x(t) = (EndPointx-StartPointx)*(N+t)+StartPointx
23     y(t) = (EndPointy-StartPointy)*(N+t)+StartPointy
24     c(t) = BLEND((t-tlo)/(thi-tlo), clo, chi)
25     a(t) = [(t-tlo)/(thi-tlo)]*(ahi-alo)+alo
26   }
27 }
```

28 Where c is the color component and a is the alpha component. t_{lo} and t_{hi} are the offsets,
 29 c_{lo} and c_{hi} are the color component values at the closest enclosing gradient stops (that is,
 30 $t_{lo} \leq t \leq t_{hi}$) and a_{lo} and a_{hi} are the alpha component values at the closest enclosing
 31 gradient stops ($t_{lo} \leq t \leq t_{hi}$).

- 32 4. If the SpreadMethod value is Reflect, the colors of points on the line defined by the
 33 StartPoint and EndPoint attributes are defined by interpolating the coordinates linearly, and
 34 each color component (such as R, G, B for sRGB and scRGB) as well as the alpha
 35 component is interpolated between the component values of the closest enclosing
 36 gradient stops:

37 For each repetition (all integers) N :

```
38 {
39   For each offset (real number)  $0 \leq t \leq 1$ :
40   {
41     If (N is EVEN)
42     {
43       x(t) = (EndPointx-StartPointx)*(N+t)+StartPointx
44       y(t) = (EndPointy-StartPointy)*(N+t)+StartPointy
45     }
46     Else
47     {
48       x(t) = (EndPointx-StartPointx)*(N+1-t)+StartPointx
49       y(t) = (EndPointy-StartPointy)*(N+1-t)+StartPointy
50     }
51   }
```

```

1      c(t) = BLEND((t-t1o)/(thi-t1o), c1o, chi)
2      a(t) = [(t-t1o)/(thi-t1o)]*(ahi-a1o)+a1o
3  }
4  }
```

5 Where c is the color component and a is the alpha component. t_{1o} and t_{hi} are the offsets,
6 c_{1o} and c_{hi} are the color component values at the closest enclosing gradient stops (that is,
7 $t_{1o} \leq t \leq t_{hi}$) and a_{1o} and a_{hi} are the alpha component values at the closest enclosing
8 gradient stops ($t_{1o} \leq t \leq t_{hi}$).

- 9 5. The colors of points not on the extended line defined by the StartPoint and EndPoint
10 attributes are the same as the color of the closest point on the line defined by the
11 StartPoint and EndPoint attributes, measured in the coordinate space as transformed by
12 the current effective render transform (including the render transform for the element
13 being filled by the linear gradient brush and the brush's transform itself).
- 14 6. Clip the resulting set of points to the intersection of the current clip geometry and the
15 path or glyphs to be filled. Both the clip and path (or glyphs) must be transformed
16 according to the current effective render transform, including the render transform for
17 the element being filled, but *not* including the transform of the linear gradient brush.

18 For purposes of the above steps, the closest enclosing gradient stops mean the gradient stops
19 that, if the relative sequencing of the gradient stop offsets in the markup order is respected,
20 are numerically closest to the interpolation point if that interpolation point were converted to an
21 offset value and inserted in a sorted fashion into the list of gradient stops. [Example: If a
22 gradient contains three gradient stops at offset values 0.0, 0.0, and 1.0, the closest enclosing
23 gradient stops for any value $0 \leq \text{value} \leq 1$ are the second gradient stop (offset 0.0) and the
24 third gradient stop (offset 1.0). end example]

25 18.3.3 Radial Gradients

26 Consumers SHOULD render an element filled with a radial gradient brush such that the
27 appearance is the same as if these steps had been followed [S11.11]:

- 28 1. The boundary of the area filled by a radial gradient brush is defined by interpolating
29 ellipses from the GradientOrigin value to the circumference of the ellipse centered at the
30 point specified by the Center attribute with radii equal to the RadiusX and RadiusY
31 attribute values (the interpolated ellipses and point being transformed by the current
32 effective render transform, including the render transform for the element being filled by
33 the radial gradient brush and the brush's transform itself). If the gradient origin is
34 outside the circumference of the ellipse specified, the effect will be as if a cone were
35 drawn, tapering to the gradient origin.
- 36 2. If the SpreadMethod value is Pad, the centers and radii of the interpolated ellipses are
37 defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute
38 value to the Center attribute value, and simultaneously linearly interpolating the radii of
39 the ellipse from 0 to the RadiusX and RadiusY attribute values:

```

40 For each offset (real number)  $0 \leq t \leq 1$ :
41 {
42   cx(t) = (Centerx-GradientOriginx)*t+GradientOriginx
43   cy(t) = (Centery-GradientOriginy)*t+GradientOriginy
44   rx(t) = RadiusX*t
45   ry(t) = RadiusY*t
46 }
```

1 The ellipses defined by the interpolation are transformed by the current effective render
2 transform, including the render transform for the element being filled by the radial
3 gradient brush and the brush's transform itself.

4 3. The colors of the points within the boundary of this shape are defined as the color of the
5 smallest interpolated ellipse containing the point. The color of an interpolated ellipse is
6 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as
7 well as the alpha component between the component values of the closest enclosing
8 gradient stops:

9 For each offset (real number) $0 \leq t \leq 1$:

```
10 {
11   c(t) = BLEND((t-t10)/(thi-t10), c10, chi)
12   a(t) = [(t-t10)/(thi-t10)]*(ahi-a10)+a10
13 }
```

14 Where t_{10} and t_{hi} are the offsets, c_{10} and c_{hi} are the color component values at the
15 closest enclosing gradient stops (that is, $t_{10} \leq t \leq t_{hi}$) and a_{10} and a_{hi} are the alpha
16 component values at the closest enclosing gradient stops ($t_{10} \leq t \leq t_{hi}$).

17 4. If the SpreadMethod value is Repeat, the centers and radii of the interpolated ellipses are
18 defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute
19 value to the Center attribute value, and simultaneously linearly interpolating the radii of
20 the ellipse from 0 to RadiusX and RadiusY attribute values:

21 For each repetition (all non-negative integers) N:

```
22 {
23   For each offset (real number)  $0 \leq t < 1$ :
24   {
25     cx(t) = (Centerx-GradientOriginx)*(N+t)+GradientOriginx
26     cy(t) = (Centery-GradientOriginy)*(N+t)+GradientOriginy
27     rx(t) = RadiusX*(N + t)
28     ry(t) = RadiusY*(N + t)
29   }
30 }
```

31 The ellipses defined by the interpolation are transformed by the current effective render
32 transform, including the render transform for the element being filled by the radial
33 gradient brush and the brush's transform itself.

34 5. The colors of the points within the boundary of this shape are defined as the color of the
35 smallest interpolated ellipse containing the point. The color of an interpolated ellipse is
36 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as
37 well as the alpha component between the component values of the closest enclosing
38 gradient stops:

39 For each repetition (all non-negative integers) N:

```
40 {
41   For each offset (real number)  $0 \leq t < 1$ :
42   {
43     c(t) = BLEND((t-t10)/(thi-t10), c10, chi)
44     a(t) = [(t-t10)/(thi-t10)]*(ahi-a10)+a10
45   }
46 }
```

47 Where t_{10} and t_{hi} are the offsets, c_{10} and c_{hi} are the color component values at the
48 closest enclosing gradient stops (that is, $t_{10} \leq t \leq t_{hi}$) and a_{10} and a_{hi} are the alpha
49 component values at the closest enclosing gradient stops ($t_{10} \leq t \leq t_{hi}$).

- 1 6. If the SpreadMethod value is Reflect, the centers and radii of the interpolated ellipses are
 2 defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute
 3 value to the Center attribute value, and simultaneously linearly interpolating the radii of
 4 the ellipse from 0 to the RadiusX and RadiusY attribute values:

```

5 For each non-negative integer N:
6 {
7   For each offset (real number)  $0 \leq t \leq 1$ :
8   {
9      $c_x(t) = (Center_x - GradientOrigin_x) * (N+t) + GradientOrigin_x$ 
10     $c_y(t) = (Center_y - GradientOrigin_y) * (N+t) + GradientOrigin_y$ 
11     $r_x(t) = RadiusX * (N+t)$ 
12     $r_y(t) = RadiusY * (N+t)$ 
13   }
14 }

```

15 The ellipses defined by the interpolation are transformed by the current effective render
 16 transform, including the render transform for the element being filled by the radial
 17 gradient brush and the brush's transform itself.

- 18 7. The colors of the points within the boundary of this shape are defined as the color of the
 19 smallest interpolated ellipse containing the point. The color of an interpolated ellipse is
 20 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as
 21 well as the alpha component between the component values of the closest enclosing
 22 gradient stops:

```

23 For each non-negative integer N:
24 {
25   For each offset (real number)  $0 \leq t \leq 1$ :
26   {
27     If N is ODD
28        $t' = 1-t$ 
29     Else
30        $t' = t$ 
31
32      $c(t) = BLEND((t' - t_{lo}) / (t_{hi} - t_{lo}), c_{lo}, c_{hi})$ 
33      $a(t) = [(t' - t_{lo}) / (t_{hi} - t_{lo})] * (a_{hi} - a_{lo}) + a_{lo}$ 
34   }
35 }

```

36 Where t_{lo} and t_{hi} are the offsets, c_{lo} and c_{hi} are the color component values at the
 37 closest enclosing gradient stops (that is, $t_{lo} \leq t \leq t_{hi}$) and a_{lo} and a_{hi} are the alpha
 38 component values at the closest enclosing gradient stops ($t_{lo} \leq t \leq t_{hi}$).

- 39 8. The colors of points outside the boundary of this shape (points which cannot be drawn by
 40 any combination of non-negative N and t) are defined as having the color and alpha
 41 defined in the gradient stop with the offset of 0.0 for radial gradients with a SpreadMethod
 42 value of Reflect and the color and alpha defined in the gradient stop with the offset of 1.0
 43 for radial gradients with a SpreadMethod value of Repeat or Pad. The colors outside of the
 44 boundary of this shape can also vary in an implementation-specific manner
 45 (see §18.3.1.2 for more details).
- 46 9. Clip the resulting set of points by the intersection of the current clip geometry and the
 47 path or glyphs to be filled. Both the clip and path (or glyphs) must be transformed
 48 according to the current effective render transform, including the render transform for
 49 the element being filled, but *not* including the transform of the radial gradient brush.

1 For purposes of the above steps, the closest enclosing gradient stops mean the gradient stops
 2 that, if the relative sequencing of the gradient stop offsets in the markup order is respected,
 3 are numerically closest to the interpolation point if that interpolation point were converted to an
 4 offset value and inserted in a sorted fashion into the list of gradient stops. [*Example*: If a
 5 gradient contains three gradient stops at offset values 0.0, 0.0, and 1.0, the closest enclosing
 6 gradient stops for any value $0 \leq \text{value} \leq 1$ are the second gradient stop (offset 0.0) and the
 7 third gradient stop (offset 1.0). *end example*]

8 **18.4 Opacity Computations**

9 Opacity is used to transparently blend two elements when rendering, also known as alpha
 10 blending. The value of the Opacity property ranges from 0.0 (fully transparent) to 1.0 (fully
 11 opaque), inclusive. Values outside of this range are invalid.

12 The opacity is applied through the following computations, assuming source and destination
 13 values are not pre-multiplied. All opacity calculations SHOULD be performed with at least 8-bit
 14 precision to provide sufficient quality for nested content [S11.18].

15 Individual pixels are blended as defined below.

16 *Table 18–2. Opacity computation symbols*

Symbol	Description
O_E	Opacity attribute of element
O_M	Alpha value at corresponding pixel position in the OpacityMask attribute value
A_S	Alpha value present in source color
R_S	Red value present in source color
G_S	Green value present in source color
B_S	Blue value present in source color
A_D	Alpha value already present in destination surface
R_D	Red value already present in destination surface
G_D	Green value already present in destination surface
B_D	Blue value already present in destination surface
A^*	Resulting Alpha value for destination surface
R^*	Resulting Red value for destination surface
G^*	Resulting Green value for destination surface
B^*	Resulting Blue value for destination surface

17 All values designated with a T subscript (as in R_{T1}) are temporary values.

18 The opacity is calculated as follows:

19 10. Multiply source alpha value with opacity value and alpha value of opacity mask.

20
$$A_{S1} = A_S * O_E * O_M$$

21 11. Pre-multiply source alpha.

22 Omit this step if the source data specifies pre-multiplied alpha (see §18.4.1 for details).

```

1   AT1 = AS1
2   RT1 = RS*AS1
3   GT1 = GS*AS1
4   BT1 = BS*AS1

```

5 12. Pre-multiply destination alpha.

6 Omit this step in consumers supporting superluminous colors (see §18.4.1 for details).

```

7   AT2 = AD
8   RT2 = RD*AD
9   GT2 = GD*AD
10  BT2 = BD*AD

```

11 13. Blend.

12 See §18.4.1 for special case handling.

```

13  AT3 = (1-AT1)*AT2+AT1
14  RT3 = (1-AT1)*RT2+RT1
15  GT3 = (1-AT1)*GT2+GT1
16  BT3 = (1-AT1)*BT2+BT1

```

17 14. Reverse pre-multiplication.

18 Omit this step in consumers supporting superluminous colors. See also §18.4.1.

19 The resulting color channel values are divided by the resulting alpha value. If the
20 resulting alpha value is 0, all color channels are set to 0 by definition, as expressed in the
21 If condition below. Each of R_{T3}, G_{T3}, B_{T3} is smaller than or equal to A_{T3} and, therefore, each
22 of the resulting R*, G*, B* is in the valid interval of [0.0,1.0] after the pre-multiplication is
23 reversed.

```

24  If AT3 = 0
25  {
26    set all A* R* G* B* to 0.
27  }
28  Else
29  {
30    A* = AT3
31    R* = RT3/AT3
32    G* = GT3/AT3
33    B* = BT3/AT3
34  }

```

35 When blending colors in a color space other than sRGB, color channels are independently
36 interpolated in a manner analogous to the RGB channel blending method described above.

1 **18.4.1 Pre-Multiplied Alpha and Superluminous Colors**

2 The alpha information in TIFF images using an ExtraSamples tag value of 1 and in Windows
3 Media Photo images using pixel formats WICPixelFormat32bppPBGRA,
4 WICPixelFormat64bppPRGBA or WICPixelFormat128bppPRGBAFloat MUST be interpreted as
5 pre-multiplied alpha information [M11.6]. In certain scenarios (such as when rendering 3D
6 scenes to a bitmap), producers MAY choose to create pre-multiplied bitmap data specifying
7 "superluminous" colors [O11.9].

8 Superluminous colors are defined as pre-multiplied color values with an alpha value smaller
9 than the individual color channel values but greater than or equal to 0.

10 The effect of composing superluminous colors on a background is similar to adding additional
11 light of the source color to the destination, as opposed to regular alpha composition which
12 works more like a colored filter. One can easily verify this statement by substituting 0 for A_{T1} in
13 step 4 of the above opacity computations, which is simplified as follows (note that R_{T1} , G_{T1} , B_{T1}
14 are not 0, because the pre-multiplication in step 2 has been skipped):

```
15      $A_{T3} = A_{T2}$ 
16      $R_{T3} = R_{T2} + R_{T1}$ 
17      $G_{T3} = G_{T2} + G_{T1}$ 
18      $B_{T3} = B_{T2} + B_{T1}$ 
```

19 Consumers supporting superluminous colors retain all temporary information in pre-multiplied
20 formats. Note, that throughout the XPS specification non-pre-multiplied alpha processing is
21 assumed. It is up to the implementer of such a consumer to identify equivalent composition and
22 rendering rules for processing in pre-multiplied space.

23 Also note, when composing superluminous colors, management of out-of-gamut colors SHOULD
24 be deferred until the result is rendered to the final target, at which point out-of-gamut colors
25 are clipped or color managed [S11.19].

26 Consumers MAY handle superluminous colors or MAY instead choose to convert pre-multiplied
27 source data containing superluminous colors to non-pre-multiplied data before composition by
28 ignoring the superluminous portion of each color channel value [O11.10], as described in the
29 following steps:

```
30     For each superluminous pixel with  $A < R$  or  $A < G$  or  $A < B$ 
31     {
32         If  $A = 0$ 
33         {
34              $R^* = 1$ 
35              $G^* = 1$ 
36              $B^* = 1$ 
37         }
38         Else
39         {
40              $A^* = A$ 
41              $R^* = \min(R/A, 1)$ 
42              $G^* = \min(G/A, 1)$ 
43              $B^* = \min(B/A, 1)$ 
44         }
45     }
```

18.5 Composition Rules

XPS Document page markup uses the painter's model with alpha channel. Composition MUST have the same effect as the application of the following rules, in sequence [M11.7]:

1. In order to render a fixed page or canvas, a surface is created to hold the drawing content as it is composed. The color and appearance of this surface SHOULD match the destination color and appearance, typically a solid white background for a fixed page or transparent for a canvas [S11.20]. An implementation MAY choose to meet this goal by always initializing this surface's alpha channel to 0.0 (transparent) and the color value to black [O11.5].
2. The fixed page or canvas represents a surface onto which child elements are drawn. The child elements are drawn in the order they appear in markup. In practice, an implementation might represent the surface by a bitmap buffer large enough to hold all the drawing content produced when the child elements are rendered.
3. The contents appearing on the surface of canvas are transformed using the affine transform specified by the RenderTransform property of the canvas. (A fixed page does not have a RenderTransform property.)
4. All child elements are rendered to the surface and clipped to the imageable area of the physical display (such as a sheet of paper) of the fixed page or according to the Clip property of a canvas. The geometry value of the canvas' Clip property is also transformed using the affine transform specified by the RenderTransform property of the canvas.
5. If a path has a Stroke and a Fill property, and also specifies Opacity or OpacityMask property values, additional composition steps must be followed:
 - a. Create a temporary canvas with the opacity, opacity mask, clip, and render transform specified by the path.
 - b. Create a copy of the original path, remove all but the Fill property from the copy, and add the copy to the temporary canvas.
 - c. Create another copy of the original path, remove all but the stroke-related properties (such as Stroke, StrokeThickness, and StrokeDashArray) from the copy, and add the copy to the temporary canvas.
 - d. Do not draw the original path.
 - e. Draw the temporary canvas, while recursively applying the composition rules.
6. If a grouping element (a <Canvas> element) has an Opacity or OpacityMask property, additional composition steps must be followed:
 - a. Create a temporary surface and set its alpha channel to 0.0 (transparent) and its color value to black.
 - b. Compose all child elements of the grouping element onto the temporary surface, while recursively applying the composition rules.
 - c. Cumulatively apply the opacity of the grouping element and opacity mask to the alpha channel of the temporary surface.
 - d. Draw the contents of the temporary surface onto the containing surface.

- 1 7. If a non-grouping element (a <Path> or <Glyphs> element) has an Opacity property, an
2 OpacityMask property, or a fill or stroke using transparency, the following additional
3 composition steps must be taken:
 - 4 a. If the element has a RenderTransform property, apply it to the element and its Clip,
5 Fill, Stroke, and OpacityMask properties, if present.
 - 6 b. Create a mask from the set of all painted pixels representing the child element (after
7 the Clip property of the element has been applied).
- 8 8. Combine the Fill or Stroke property with the OpacityMask and the Opacity property and
9 apply to the surface through the computed mask. For more information, see §14.1.

10 The behavior that results from this process is:

- 11 • Opacity is not applied cumulatively to self-overlapping areas created when rendering an
12 individual <Glyphs> element.
- 13 • Opacity is not applied cumulatively to self-overlapping areas created by <PathFigure>
14 elements within the same path (see Example 18–1).
- 15 • Opacity is not applied cumulatively if the border of a path has self-intersections. When
16 the border of a path is stroked, the area of the path is filled by first applying the brush
17 specified by the Fill property. After filling the area, the border is drawn using the stroke-
18 related properties including the brush specified by the Stroke property, with half the
19 stroke width extending outside the filled area and half extending inside (see Example
20 18–2). If the path has self-intersections, the opacity is not accumulated.
- 21 • The color of the stroke and the color of the filled area are combined on the inside half of
22 a stroked border (overlapping the filled area of the path) if the brush specified by the
23 Stroke property is transparent.
- 24 • If a path that has a stroked border has an opacity of less than 1.0 or an opacity mask,
25 the path (filled area and stroked border) is first rendered onto a temporary surface using
26 an opacity of 1.0 and no opacity mask (while preserving any transparency of the fill or
27 the stroked border themselves), and the resulting figure is drawn onto the background
28 using the specified opacity and opacity mask (see Example 18–3).

29 **18.5.1 Optimization Guidelines**

30 The composition rules above describe the behavior of an ideal implementation. Practical
31 implementations can optimize the processing of the composition rules according to the following
32 guidelines:

- 33 9. If all elements on a canvas and the canvas itself are opaque (an opacity of 1.0) and
34 parent or ancestor <Canvas> elements are also opaque, the elements MAY be drawn
35 directly to the containing fixed page (or canvas), provided all render transform and clip
36 values are observed [O11.12].
- 37 10. If an element is fully transparent (an opacity of 0.0), it MAY be skipped [O11.13].
- 38 11. If a canvas has an opacity of 0.0, it and all of its child and descendant elements MAY be
39 skipped [O11.14].
- 40 12. If a canvas has a Clip property with no contained area, the canvas and all of its child and
41 descendant elements MAY be skipped [O11.15].
- 42 13. When creating a temporary surface, a consumer MAY further restrict the size of the
43 temporary surface by the effective extent of the geometry specified by the Clip property
44 of the canvas [O11.16].

1 14. A consumer MAY use methods to achieve transparency other than creating a temporary
 2 surface [O11.17]. Such methods MAY include planar mapping (that is, computation of
 3 intersections of transparent elements and resulting colors) [O11.17].

4 **18.5.2 Composition Examples**

5 The following examples illustrate the composition rules described above.

6 *Example 18–1. Path opacity behavior for overlapping path figures*

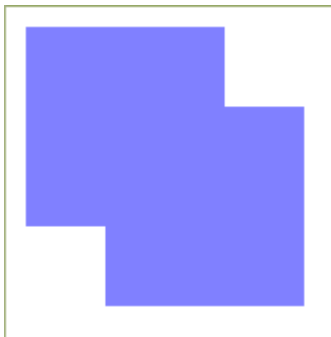
7 In the following markup, opacity is not applied cumulatively to self-overlapping areas created
 8 by path figures within the same path.

```

9     <Path Opacity="0.5">
10     <Path.Fill>
11         <SolidColorBrush Color="#0000FF" />
12     </Path.Fill>
13     <Path.Data>
14         <PathGeometry FillRule="NonZero">
15             <PathFigure StartPoint="10,10" IsClosed="true">
16                 <PolyLineSegment Points="110,10 110,110 10,110 10,10" />
17             </PathFigure>
18             <PathFigure StartPoint="50,50" IsClosed="true">
19                 <PolyLineSegment Points="150,50 150,150 50,150 50,50" />
20             </PathFigure>
21         </PathGeometry>
22     </Path.Data>
23 </Path>

```

24 This markup is rendered as follows:



25

26 *end example]*

27 *Example 18–2. Opacity behavior of path stroke intersections*

28 In the following markup, opacity is not applied cumulatively if the border of a path has self-
 29 intersections.

```

30     <Path Stroke="#80FF0000" StrokeThickness="10">
31     <Path.Fill>
32         <SolidColorBrush Color="#0000FF" />
33     </Path.Fill>
34     <Path.Data>
35         <PathGeometry FillRule="NonZero">
36             <PathFigure StartPoint="20,20" IsClosed="true">

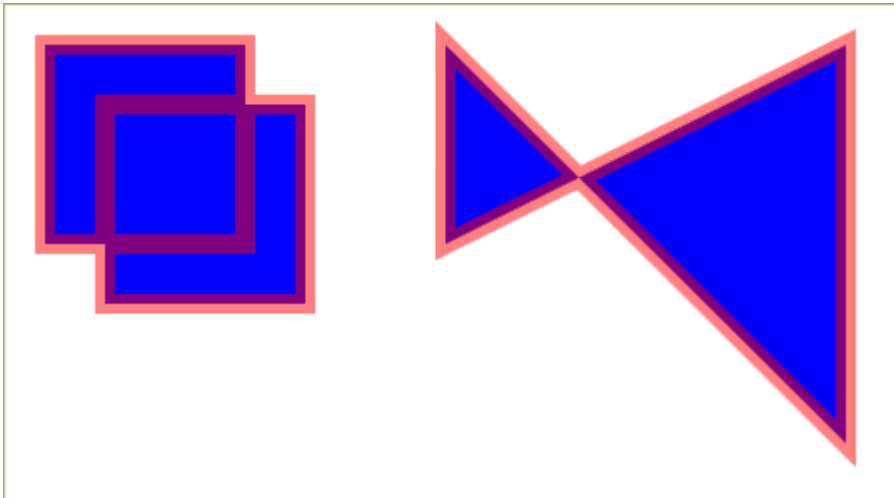
```

```

1         <PolyLineSegment Points="120,20 120,120 20,120 20,20" />
2     </PathFigure>
3     <PathFigure StartPoint="50,50" IsClosed="true">
4         <PolyLineSegment Points="150,50 150,150 50,150 50,50" />
5     </PathFigure>
6 </PathGeometry>
7 </Path.Data>
8 </Path>
9 <Path Stroke="#80FF0000" StrokeThickness="10" StrokeMiterLimit="10">
10    <Path.Fill>
11        <SolidColorBrush Color="#0000FF" />
12    </Path.Fill>
13    <Path.Data>
14        <PathGeometry>
15            <PathFigure StartPoint="220,20" IsClosed="true">
16                <PolyLineSegment Points="420,220 420,20 220,120" />
17            </PathFigure>
18        </PathGeometry>
19    </Path.Data>
20 </Path>

```

21 This markup is rendered as follows:



22

23 *end example]*

24 *Example 18-3. Opacity behavior of paths with stroked edges*

25 The following markup describes a path with a stroked border and an opacity of less than 1.0:

```

26 <Path>
27     <Path.Fill>
28         <SolidColorBrush Color="#7F7F7F" />
29     </Path.Fill>
30     <Path.Data>
31         <PathGeometry>
32             <PathFigure StartPoint="0,110" IsClosed="true">
33                 <PolyLineSegment Points="450,110 450,210 0,210" />
34             </PathFigure>
35         </PathGeometry>

```

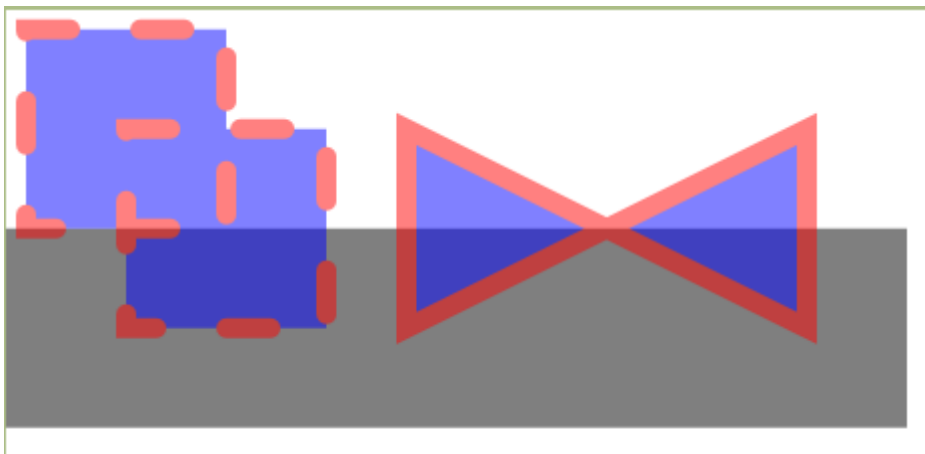


```

1     </Path.Data>
2 </Path>
3 <Path
4     Stroke="#FF0000"
5     StrokeThickness="10"
6     StrokeDashArray="2.2 3.5"
7     StrokeDashCap="Round"
8     Opacity="0.5">
9     <Path.Fill>
10    <SolidColorBrush Color="#0000FF" />
11 </Path.Fill>
12 <Path.Data>
13    <PathGeometry FillRule="NonZero">
14      <PathFigure StartPoint="10,10" IsClosed="true">
15        <PolyLineSegment Points="110,10 110,110 10,110 10,10" />
16      </PathFigure>
17      <PathFigure StartPoint="60,60" IsClosed="true">
18        <PolyLineSegment Points="160,60 160,160 60,160 60,60" />
19      </PathFigure>
20    </PathGeometry>
21 </Path.Data>
22 </Path>
23 <Path Stroke="#FF0000" StrokeThickness="10" Opacity="0.5">
24   <Path.Fill>
25     <SolidColorBrush Color="#0000FF" />
26   </Path.Fill>
27   <Path.Data>
28     <PathGeometry>
29       <PathFigure StartPoint="200,60" IsClosed="true">
30         <PolyLineSegment Points="400,160 400,60 200,160" />
31       </PathFigure>
32     </PathGeometry>
33   </Path.Data>
34 </Path>

```

35 This markup is rendered as follows:



36

37 *end example]*

18.6 Stroke Rendering

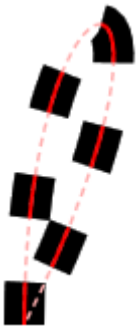
Strokes follow the contours of each segment in a path figure, as specified by the various stroke-related properties.

Contours and dashes SHOULD be rendered so that they have the same appearance as if rendered by sweeping the complete length of the contour or dash with a line segment that is perpendicular to the contour and extends with half its length to each side of the contour. All points covered by the sweep of this perpendicular line are part of the dash or contour [S11.21].

By using this sweeping definition, extreme curvatures can result in line and dash ends that are not flat when specified as flat. If any caps other than flat are specified, the caps are added to the start and end of the stroked contour or dash in the orientation of the first and last position of the line segment used for sweeping. Any render transform is applied after this step.

[*Note*: Using this definition, any geometry that is less than the value of the stroke thickness across will produce a filled area between these lines if no dashes are employed, or overlapping dashes when they are. *end note*]

Figure 18–1. Extreme curvatures and dash rendering



18.6.1 Stroke Edge Parallelization

Consumers SHOULD ensure that parallel edges of strokes appear parallel [S11.22]. Consumers can choose a suitable method to achieve this goal. [*Example*: Such methods might include anti-aliasing, sub-pixel masking, or appropriate rounding of device coordinates. *end example*]

18.6.2 Phase Control

Consumers SHOULD produce a visually consistent appearance of stroke thickness for thin lines, regardless of their orientation or how they fit on the device pixel grid [S11.23].

18.6.3 Symmetry of Stroke Drawing Algorithms

Consumers SHOULD select line and curve drawing algorithms that behave symmetrically and result in the same set of device pixels being drawn regardless of the direction of the line or curve (start point and end point exchanged) [S11.24]. In other words, a line from 0,0 to 102,50 should result in the same pixel set as a line from 102,50 to 0,0.

1 **18.6.4 Rules for Dash Cap Rendering**

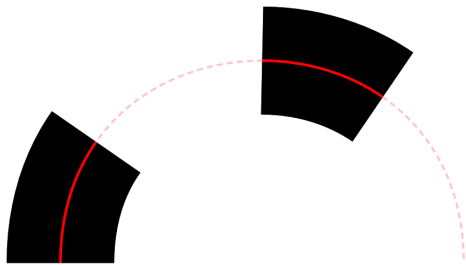
2 The appearance of dash caps is controlled by the StrokeDashCap attribute. Valid values are Flat,
3 Square, Round, and Triangle. [The StrokeDashCap attribute is ignored for paths that have no
4 StrokedDashArray attribute or that have a StrokedDashArray attribute with value 0,0.](#)

5 **18.6.4.1 Flat Dash Caps**

6 The effective render transform of the path being stroked is used to transform the control points
7 of the contour of the dash.

8 The length of the dash is the approximate distance on the curve between the two intersections
9 of the flat lines ending the dash and the contour of the shape. The distance from the end of one
10 dash to the start of the next dash is the specified dash gap length. Dashes with a length greater
11 than 0 are drawn, and degenerate dashes with a length of 0 are not drawn.

12 *Figure 18-2. Flat dash caps*



13 **18.6.4.2 Square Dash Caps**

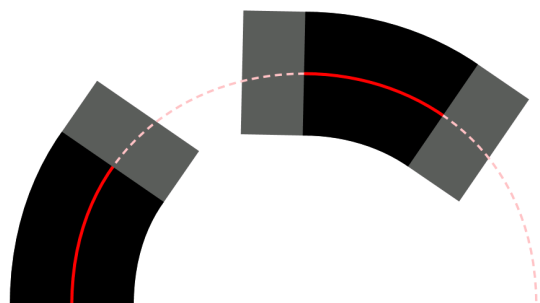
14 The effective render transform of the path being stroked is used to transform the control points
15 of the contour of the dash.

16 The length of the dash is the approximate distance on the curve between the two *contour*
17 *intersection points*, that is, the intersection of the flat line ending the dash (without the square
18 caps attached) and the contour of the shape.

19 The caps are drawn as half-squares attached to the ends of the dash. The boundaries of the
20 square caps are not curved to follow the contour, but are transformed using the effective
21 render transform.

22 The distance between the contour intersection points of consecutive dashes is the specified
23 dash gap length. Degenerate dashes with a length of 0 are drawn as squares.

24 *Figure 18-3. Square dash caps*



1 **18.6.4.3 Round Dash Caps**

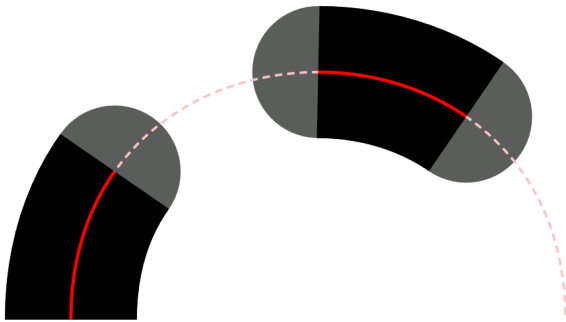
2 The effective render transform of the path being stroked is used to transform the control points
3 of the contour of the dash.

4 The length of the dash is the approximate distance on the curve between the two contour
5 intersection points, that is, the intersection of the flat line ending the dash (without the round
6 caps attached) and the contour of the shape.

7 The caps are drawn as half-circles attached to the ends of the dash. The boundaries of the
8 round caps are not distorted to follow the contour, but are transformed using the effective
9 render transform.

10 The distance between the contour intersection points of consecutive dashes is the specified
11 dash gap length. Degenerate dashes with a length of 0 are drawn as circles.

12 *Figure 18–4. Round dash caps*



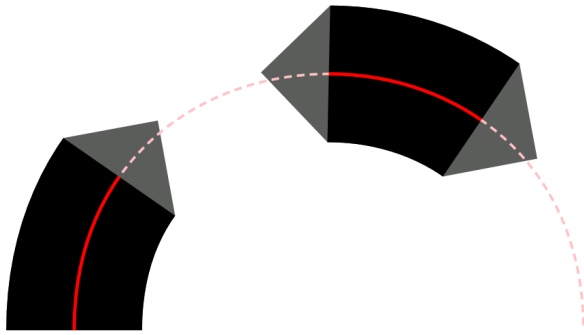
14 **18.6.4.4 Triangular Dash Caps**

15 The effective render transform of the path being stroked is used to transform the control points
16 of the contour of the dash.

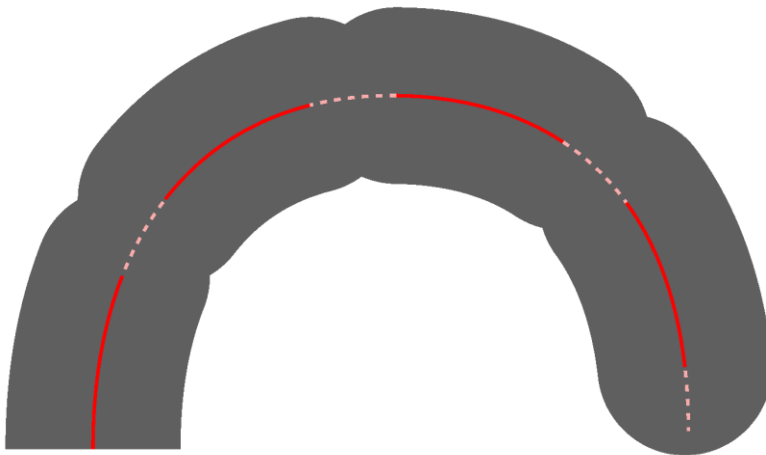
17 The length of the dash is the approximate distance on the curve between the two contour
18 intersection points, that is, the intersection of the flat line ending the dash (without the
19 triangular caps attached) and the contour of the shape.

20 The caps are drawn as triangles attached with their base to the ends of the dash. The
21 boundaries of the triangular caps are not distorted to follow the contour, but are transformed
22 using the effective render transform. The height of the triangles is half of the stroke width.

23 The distance between the contour intersection points of consecutive dashes is the specified
24 dash gap length. Degenerate dashes with a length of 0 are drawn as diamonds.

1 *Figure 18-5. Triangular dash caps*2
3 **18.6.4.5 Overlapping Dashes**

4 It is possible to specify dash sequences with overlapping dash caps. In this circumstance, the
5 union of the dash segments (inclusive of dash caps), is used as a mask through which the
6 brush is applied as illustrated in Figure 18-6 with a stroke dash cap value of Round.

7 *Figure 18-6. Overlapping dash segments*8
9 **18.6.4.6 Extreme Degenerate Dash Case**

10 [The previous subclauses include a description of the behaviour for degenerate dashes of zero](#)
11 [length, with non-zero gaps, for each dash cap shape.](#)

12 [Producers SHOULD NOT create files containing the extreme degenerate case of](#)
13 [StrokeDashArray = "0 0". Such lines SHOULD be rendered as a solid line \[S11.32\].](#)

14 **18.6.5 Rules for Line Cap Rendering**

15 The appearance of line caps is controlled by the StrokeStartLineCap and StrokeEndLineCap
16 attribute. Valid values are Flat, Square, Triangle, and Round. Every start line cap can be used in
17 combination with any end line cap. ~~Line caps are not drawn for closed paths.~~ [Line caps only ever](#)
18 [appear at the start and end of an open path, and then only if the initial/final segment is](#)
19 [stroked.](#)

20 The rules for line caps on curved lines are analogous to the rules for dash cap rendering. For
21 more information, see §18.6 and §18.6.4.

1 *Figure 18-7. Flat start line cap, flat end line cap*



2
3 *Figure 18-8. Square start line cap, square end line cap*



4
5 *Figure 18-9. Triangular start line cap, triangular end line cap*



6
7 *Figure 18-10. Round start line cap, round end line cap*



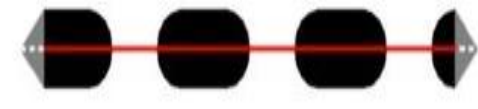
8
9 **18.6.6 Line Caps for Dashed Strokes**

10 If the start point of a stroke is within a dash or touches the start or end of a dash, a start line
11 cap is appended to the stroke. Similarly, if the end point of a stroke is within a dash or touches
12 the start or end of a dash, an end line cap is appended to the stroke.

13 *Figure 18-11. Stroke start or end point within a dash [for flat dash caps](#)*



14
15 *Figure 18-12. Stroke start or end point within a dash [for non-flat dash caps](#)*



16
17 [Note: Because the right-most line cap begins at the point exactly coincident with the start of
18 the next dash in the sequence, it is rendered. *end note*]

19 However, if the start point of a stroke is within a gap (as can result from a StrokeDashOffset
20 attribute), no start line cap is appended to the stroke. If the end point of a stroke is within a
21 gap, no end line cap is appended to the stroke.

1 *Figure 18–13. Stroke start or end point within a gap [for flat dash caps](#)*



2

3 *Figure 18–14. Stroke start or end point within a gap [for not-flat dash caps](#)*



4

5 [\[Note: Differences in precision in the calculation of coordinates can lead to differing output](#)
 6 [between consumers depending on whether they determine that the start or end point of a](#)
 7 [stroke exactly touches the start or end point of a dash. *end note*\]](#)

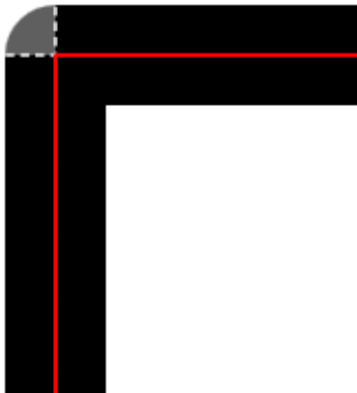
8 **18.6.7 Rules for Line Join Rendering**

9 The appearance of line joins is controlled by the StrokeLineJoin attribute. Valid values are Round,
 10 Bevel, and Miter.

11 **18.6.7.1 Round Line Joins**

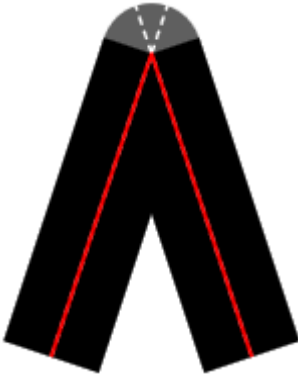
12 A StrokeLineJoin attribute value of Round indicates that the outer corner of the joined lines
 13 should be filled by enclosing the rounded region with its center point at the point of intersection
 14 between the two lines and a radius of one-half the stroke thickness value.

15 *Figure 18–15. Round line join with right angle*



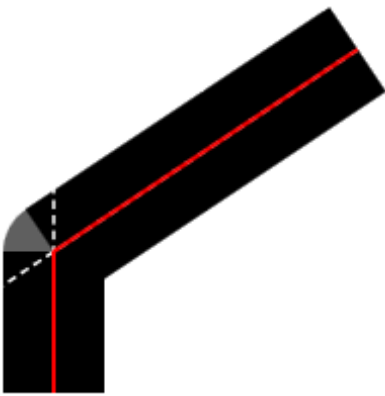
16

1 *Figure 18-16. Round line join with acute angle*



2

3 *Figure 18-17. Round line join with obtuse angle*

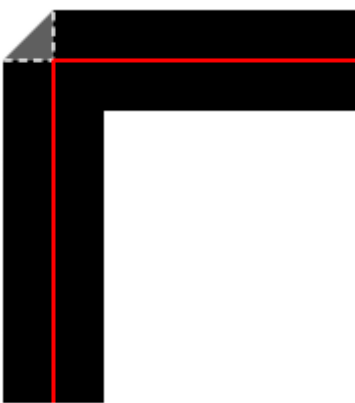


4

5 **18.6.7.2 Beveled Line Joins**

6 A StrokeLineJoin attribute value of Bevel indicates that the outer corner of the joined lines should
7 be filled by enclosing the triangular region of the corner with a straight line between the outer
8 corners of each stroke.

9 *Figure 18-18. Beveled line join with right angle*



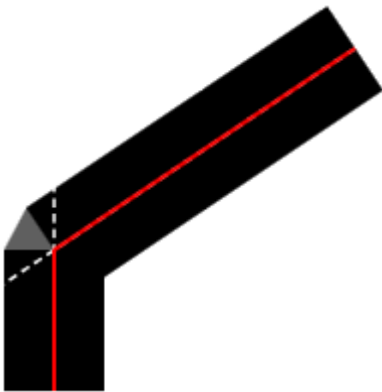
10

1 *Figure 18–19. Beveled line join with acute angle*



2

3 *Figure 18–20. Beveled line join with obtuse angle*



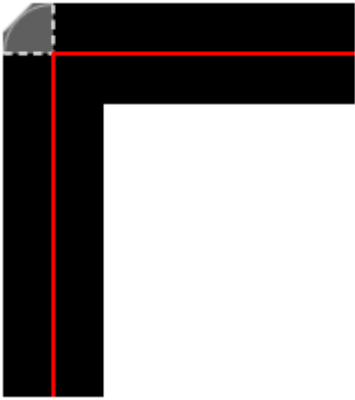
4

5 **18.6.7.3 Mitered Line Joins**

6 If the `StrokeLineJoin` attribute value is `Miter`, the value of the `StrokeMiterLimit` attribute value is
 7 also used for rendering these joins. A `StrokeLineJoin` value of `Miter` indicates that the region to
 8 be filled includes the intersection of the strokes projected to infinity, and then clipped at a
 9 specific distance. The intersection of the strokes is clipped at a line perpendicular to the bisector
 10 of the angle between the strokes, at the distance equal to the stroke miter limit value multiplied
 11 by half the stroke thickness value.

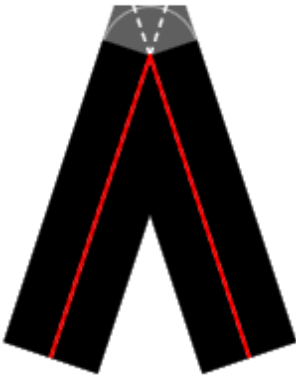
12 When drawing mitered line joins, the presence of one or more degenerate line segments
 13 between the non-degenerate line segments to be joined results in a mitered line join of only the
 14 two non-degenerate line segments with an implied `StrokeMiterLimit` attribute value of 1.0.

1 *Figure 18–21. Mitered line join with right angle and miter limit of 1.0*



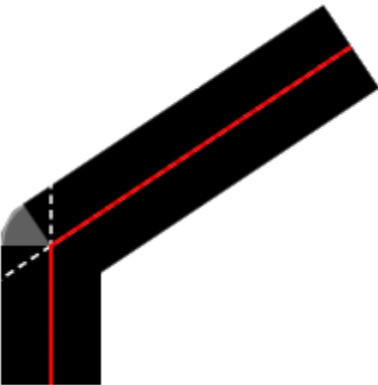
2

3 *Figure 18–22. Mitered line join with acute angle and miter limit of 1.0*



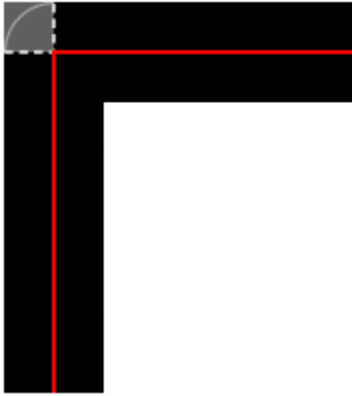
4

5 *Figure 18–23. Mitered line join with obtuse angle and miter limit of 1.0*



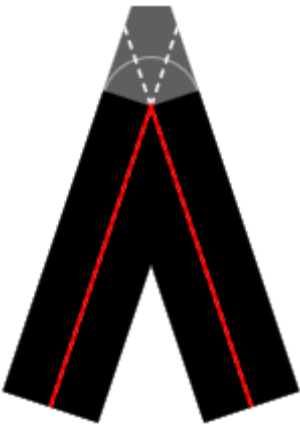
6

1 *Figure 18–24. Mitered line join with right angle and miter limit of 2.0*



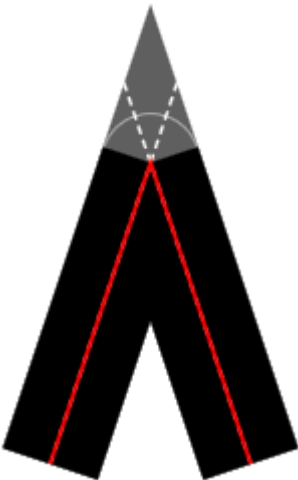
2

3 *Figure 18–25. Mitered line join with acute angle and miter limit of 2.0*



4

5 *Figure 18–26. Mitered line join with acute angle and miter limit of 10.0*



6

7 **18.6.8 Rules for Degenerate Line and Curve Segments**

8 Degenerate line segments (that is, where the start point and end point coincide) are not drawn.

1 Degenerate curve segments (where the start point, end point, and all control points coincide)
 2 are not drawn, ~~regardless of the stroke start line cap or stroke end line cap specified.~~

3 If an open degenerate path (formed from degenerate line or curve segments) with non-flat
 4 start cap and/or non-flat end line cap is stroked, only the start line cap and/or end line cap is
 5 drawn, in the x direction relative to the current effective render transform (that is, as if a
 6 segment were drawn from x,y to $x+d,y$, with $d \rightarrow 0$).

7 If a closed degenerate path (formed from degenerate line or curve segments) is stroked, a
 8 circular dot with a diameter of the stroke thickness is drawn instead.

9 If the current render transform is an invertible matrix, consumers SHOULD perform
 10 computations on poly line segments and poly Bézier segments with sufficient accuracy to avoid
 11 producing zero-length segments [S11.25].

12 **18.6.9 Stroking and Fill Rule**

13 Stroking a path is independent of the fill rule. The fill rule affects the filled area only.

14 **18.6.10 Mixing Stroked and Non-Stroked Segments**

15 When a path figure contains multiple segments and one or more of the segments has an
 16 IsStroked value of false, the phase for dashes starts anew with the next stroked segment,
 17 including application of the dash offset.

18 When a segment [of a dashed path](#) is stroked and the subsequent segment has an IsStroked
 19 value of false, thus causing a dash to be truncated, the dash cap is drawn for both ends of the
 20 truncated dash, exactly as it would for a non-truncated dash. [For the case of a closed dashed
 21 path, this rule also applies to dashes exposed at the beginning or end of the path by an
 22 unstroked final or initial segment respectively.](#)

23 **18.6.11 Stroke Behavior with Multiple Path Figures**

24 When a geometry containing multiple path figures is stroked, the phase for dashes (including
 25 application of the dash offsets) starts anew with each new path figure.

26 In general, for any path geometry, each path figure is drawn independently of every other path
 27 figure, so the dash array is reset for each. Dashes are also reset after every unstroked
 28 segment.

29 **18.6.12 Consistent Nominal Stroke Width**

30 For certain scenarios, it is desirable for producers to generate documents targeted at specific
 31 aliasing consumers with particular lines in the document indicated as hairlines or consistent-
 32 width strokes. The following recommendation allows these producers and consumers to handle
 33 these strokes consistently.

34 Producers MAY generate a <Path> element intended to be treated as having a consistent
 35 nominal stroke width by specifying the StrokeDashArray attribute and by specifying a
 36 StrokeDashOffset attribute value less than -1.0 times the sum of all the numbers in the
 37 StrokeDashArray attribute value [O11.25].

38 For a solid line, the producer would set the StrokeDashArray to the value "1 0" and the
 39 StrokeDashOffset to a value such as "-2". The "-2" value fulfills the restriction on the
 40 StrokeDashOffset value in a numerically stable manner, and the phase of the dash pattern is

1 identical to a StrokeDashOffset value of "0". Values less than "-2" can be used to specify a shifted
2 phase of the dash pattern.

3 A stroke using the consistent nominal stroke width convention SHOULD be rendered with a
4 width consistent with other strokes using the convention that have the same StrokeThickness
5 attribute value, and consumers aware of this convention SHOULD render such a stroke no
6 thinner than the thinnest visible line that consumer supports without dropouts [S11.31]. See
7 §11.1.4, for further considerations for rendering thin lines.

8 **18.7 Brushes and Images**

9 Images require the following special considerations for scaling and tile placement.

10 **18.7.1 Small Tiles**

11 Tiles for visual brushes and image brushes can be specified with a viewport width or height of a
12 few device pixels, or even less than a single device pixel in size.

13 If both width and height are nearly zero, implementations SHOULD average the color values of
14 the brush contents, resulting in a constant-color brush [S11.26]. [Example:

- 15 • A visual brush or image brush that contains a blue and white checkerboard pattern
16 results in a solid light-blue fill as either the width or the height value approaches 0.0.
- 17 • A visual brush or image brush whose viewbox is constant-colored produces a constant-
18 colored brush regardless of the width and height values of the viewport.

19 *end example]*

20 If only one of the width and height values is nearly zero, the brush should be constant-colored
21 along lines parallel to the narrow side of the viewport. For cases such as these,
22 implementations MAY differ [O11.21]. Producers SHOULD avoid producing such extreme cases
23 and SHOULD NOT rely on any specific behavior when they do [S11.27].

24 **18.7.2 Image Scaling**

25 Source sampling SHOULD be done from the center of the pixel and should be mapped to the
26 center of the pixel in the device-space [S11.28]. With one extent of the viewbox zero, sampling
27 SHOULD be done along a line parallel to the non-zero side [S11.28]. With both extents of the
28 viewbox zero, a point sample SHOULD be taken [S11.28].

29 When up-sampling an image presented at a lower resolution than the device resolution, bilinear
30 filtering SHOULD be used [S11.29]. The precise source coordinates as specified by the viewbox
31 MUST be used to place the up-sampled image tile, which is equivalent to using fractional pixels
32 of the original source image [M11.8].

33 When down-sampling, at least a bilinear filter SHOULD be used [S11.30]. Consumers MAY
34 choose to implement a more sophisticated algorithm, such as a Fant scaler, to prevent aliasing
35 artifacts [O11.22].

36 **18.7.3 Tile Placement**

37 Consumers MUST precisely position the tiles specified by the image brush and visual brush. If
38 the specified values result in fractional device pixels, the consumer MUST calculate a running
39 placement-error delta and adjust the placement of the next tile where the delta reaches a full

1 device pixel in order to keep the tiles from being increasingly out of phase as the expanse of
2 the path is filled [M11.9]. Consumers MAY choose any technique desired to achieve this
3 requirement, such as linear filtering for seams, stretching of the tile (up-sampling or down-
4 sampling), or pre-computing multiple tiles and adjusting behavior according to how the tiles fit
5 on a grid [O11.23].

6 **18.7.4 Tiling Transparent Visual Brushes and Image Brushes**

7 The contents of a visual brush's Visual property are first rendered to a temporary work canvas
8 with an opacity of 0.0. The viewBox of the visual brush defines the tile or portion of the
9 temporary canvas that is copied onto the specified geometry, stroke, or text. Likewise, an
10 image specified by an image brush is also copied to a temporary work canvas. The viewBox also
11 defines the tile for an image brush. In either case, the work canvas is scaled to properly match
12 the edges of the tile to the size specified by the viewport.

13 Each pixel of the resultant tile is separately blended with the background of its destination,
14 using the alpha of each pixel. This process is repeated for each tile replication, while respecting
15 the TileMode attribute value, although the temporary work canvas MAY be re-used [O11.24].

16

19. Elements

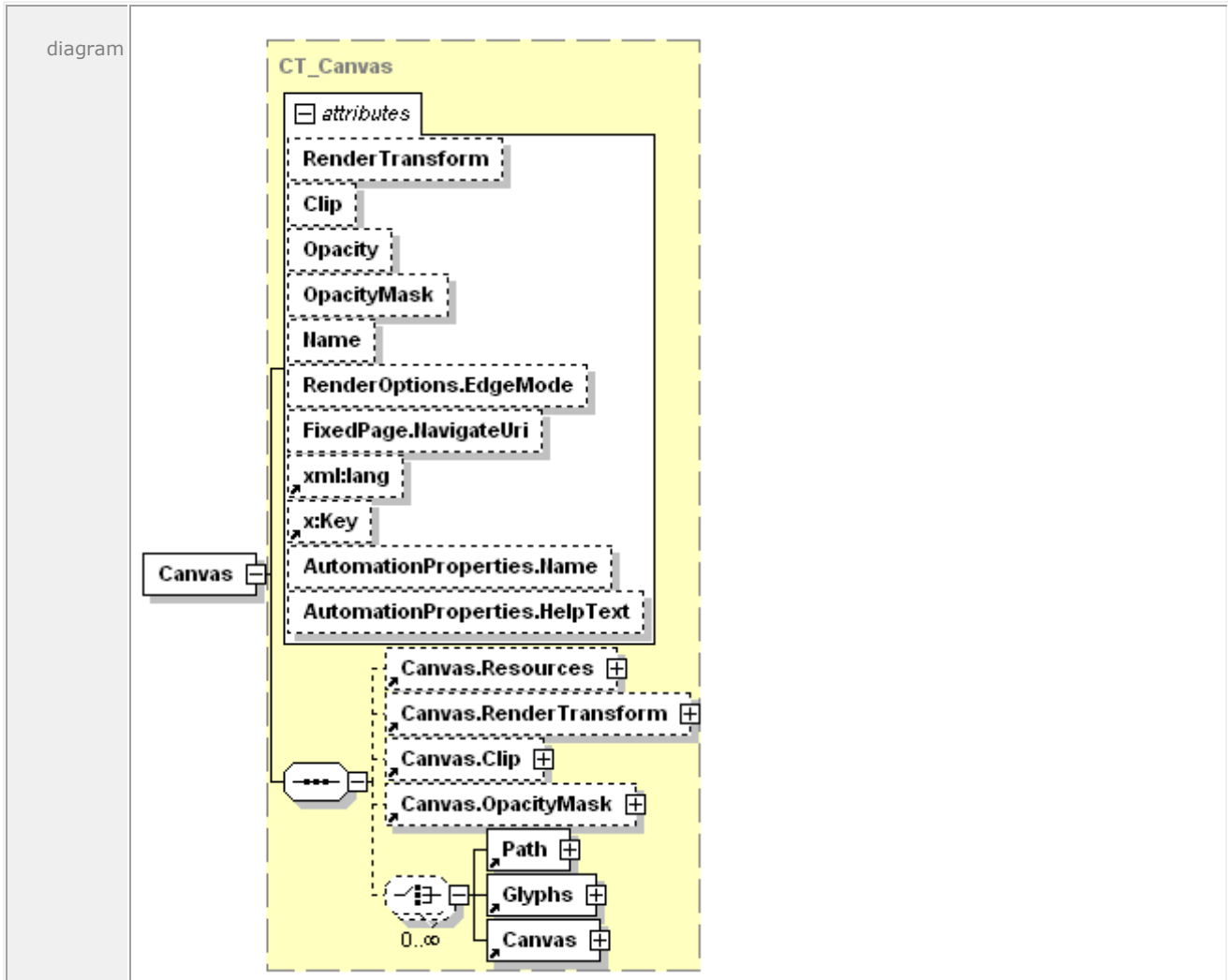
19.1 ArcSegment

element **ArcSegment**

<p>diagram</p>																																															
<p>attributes</p>	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Point</td> <td><u>ST_Point</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the endpoint of the elliptical arc.</td> </tr> <tr> <td>Size</td> <td><u>ST_PointGE0</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the x and y radius of the elliptical arc as an x,y pair.</td> </tr> <tr> <td>RotationAngle</td> <td><u>ST_Double</u></td> <td>required</td> <td></td> <td></td> <td>Indicates how the ellipse is rotated relative to the current coordinate system.</td> </tr> <tr> <td>IsLargeArc</td> <td><u>ST_Boolean</u></td> <td>required</td> <td></td> <td></td> <td>Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.</td> </tr> <tr> <td>SweepDirection</td> <td><u>ST_SweepDirection</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.</td> </tr> <tr> <td>IsStroked</td> <td><u>ST_Boolean</u></td> <td></td> <td>true</td> <td></td> <td>Specifies whether the stroke for this segment of the path is drawn. Can be true or false.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Point	<u>ST_Point</u>	required			Specifies the endpoint of the elliptical arc.	Size	<u>ST_PointGE0</u>	required			Specifies the x and y radius of the elliptical arc as an x,y pair.	RotationAngle	<u>ST_Double</u>	required			Indicates how the ellipse is rotated relative to the current coordinate system.	IsLargeArc	<u>ST_Boolean</u>	required			Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.	SweepDirection	<u>ST_SweepDirection</u>	required			Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.				
Name	Type	Use	Default	Fixed	Annotation																																										
Point	<u>ST_Point</u>	required			Specifies the endpoint of the elliptical arc.																																										
Size	<u>ST_PointGE0</u>	required			Specifies the x and y radius of the elliptical arc as an x,y pair.																																										
RotationAngle	<u>ST_Double</u>	required			Indicates how the ellipse is rotated relative to the current coordinate system.																																										
IsLargeArc	<u>ST_Boolean</u>	required			Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.																																										
SweepDirection	<u>ST_SweepDirection</u>	required			Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.																																										
IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.																																										
<p>annotation</p>	<p>Represents an elliptical arc between two points.</p>																																														

1 **19.2 Canvas**

2 element **Canvas**



attributes	Name	Type	Use	Default	Fixed	Annotation
	RenderTransform	ST_RscRefMatrix				Establishes a new coordinate frame for the child and descendant elements of the canvas, such as another canvas. Also affects clip and opacity mask.
	Clip	ST_RscRefAbbrGeomF				Limits the rendered region of the element.
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the canvas. Values range from 0 (fully

					transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
OpacityMask	<u>ST_RscRef</u>				Specifies a mask of alpha values that is applied to the canvas in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
Name	<u>ST_Name</u>				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
RenderOptions.EdgeMode	<u>ST_EdgeMode</u>				Controls how edges of paths within the canvas are rendered. The only valid value is Aliased. Omitting this attribute causes the edges to be rendered in the consumer's default manner.
FixedPage.NavigateUri	xs:anyURI				Associates a hyperlink URI with the element. May be a relative reference or a URI that addresses a resource that is internal to or external to the package.
xml:lang					Specifies the default language used for the current element and for any child or descendant elements. The language is specified according to RFC 3066.
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M3.20].
AutomationProperties.Name	xs:string				A brief description of the <Canvas> contents for

						accessibility purposes, particularly if filled with a set of vector graphics and text elements intended to comprise a single vector graphic.
	AutomationProperties.HelpText	xs:string				A detailed description of the <Canvas> contents for accessibility purposes, particularly if filled with a set of graphics and text elements intended to comprise a single vector graphic.
annotation	Groups <FixedPage> descendant elements together.					

1 For more information, see §10.4.

2 19.3 Canvas.Clip

3 element **Canvas.Clip**

diagram	
annotation	Limits the rendered region of the element.

4 For more information, see §14.3.

5 19.4 Canvas.OpacityMask

6 element **Canvas.OpacityMask**

diagram	
annotation	Specifies a mask of alpha values that is applied to the canvas in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.

1 For more information, see §14.5.1.

2 **19.5 Canvas.RenderTransform**

3 element **Canvas.RenderTransform**

diagram	
annotation	Establishes a new coordinate frame for the child and descendant elements of the canvas, such as another canvas. Also affects clip and opacity mask.

4 For more information, see §14.4.

5 **19.6 Canvas.Resources**

6 element **Canvas.Resources**

diagram	
annotation	Contains the resource dictionary for the <Canvas> element.

7 For more information, see §14.2.

8 **19.7 Discard**

9 element **Discard**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	SentinelPage	xs:anyURI	required			The first fixed page that no longer needs the identified resource in order to be processed.
	Target	xs:anyURI	required			The resource that can be safely discarded.

annotation	Identifies a resource that can be safely discarded by a resource-constrained consumer.
------------	--

1 For more information, see §17.1.4.1.2.

2 **19.8 DiscardControl**

3 element **DiscardControl**

diagram	
annotation	Contains a list of resources that are safe for a consumer to discard.

4 For more information, see §17.1.4.1.1.

5 **19.9 DocumentOutline**

6 element **DocumentOutline**

diagram													
attributes	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Name</th> <th style="width: 10%;">Type</th> <th style="width: 15%;">Use</th> <th style="width: 15%;">Default</th> <th style="width: 10%;">Fixed</th> <th style="width: 35%;">Annotation</th> </tr> </thead> <tbody> <tr> <td>xml:lang</td> <td></td> <td>required</td> <td></td> <td></td> <td>This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	xml:lang		required			This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.
Name	Type	Use	Default	Fixed	Annotation								
xml:lang		required			This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.								
annotation	Specifies a list of meaningful indices into the XPS Document, similar to a table of contents, or to external URIs, such as web addresses.												

7 For more information, see §16.1.1.3.

8 **19.10 DocumentReference**

9 element **DocumentReference**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Source	xs:anyURI	required			Specifies the URI of the fixed document content. The specified URI MUST refer to a FixedDocument part within the XPS Document [M3.2].
annotation	Contains a reference to a FixedDocument part.					

1 For more information, see §10.1.1.

2 19.11 DocumentStructure

3 element **DocumentStructure**

diagram						
annotation	The root element of the DocumentStructure part.					

4 For more information, see §16.1.1.1.

5 19.12 DocumentStructure.Outline

6 element **DocumentStructure.Outline**

diagram						
annotation	Contains a structured document outline that provides a list of links into the document contents or external sites.					

8 For more information see §16.1.1.2.

1 **19.13 FigureStructure**

2 element **FigureStructure**

diagram	
annotation	Groups the named elements that constitute a single drawing or diagram.

3 For more information, see §16.1.2.12.

4 **19.14 FixedDocument**

5 element **FixedDocument**

diagram	
annotation	Binds an ordered sequence of fixed pages together into a single multi-page document.

6 For more information, see §10.2.

7 **19.15 FixedDocumentSequence**

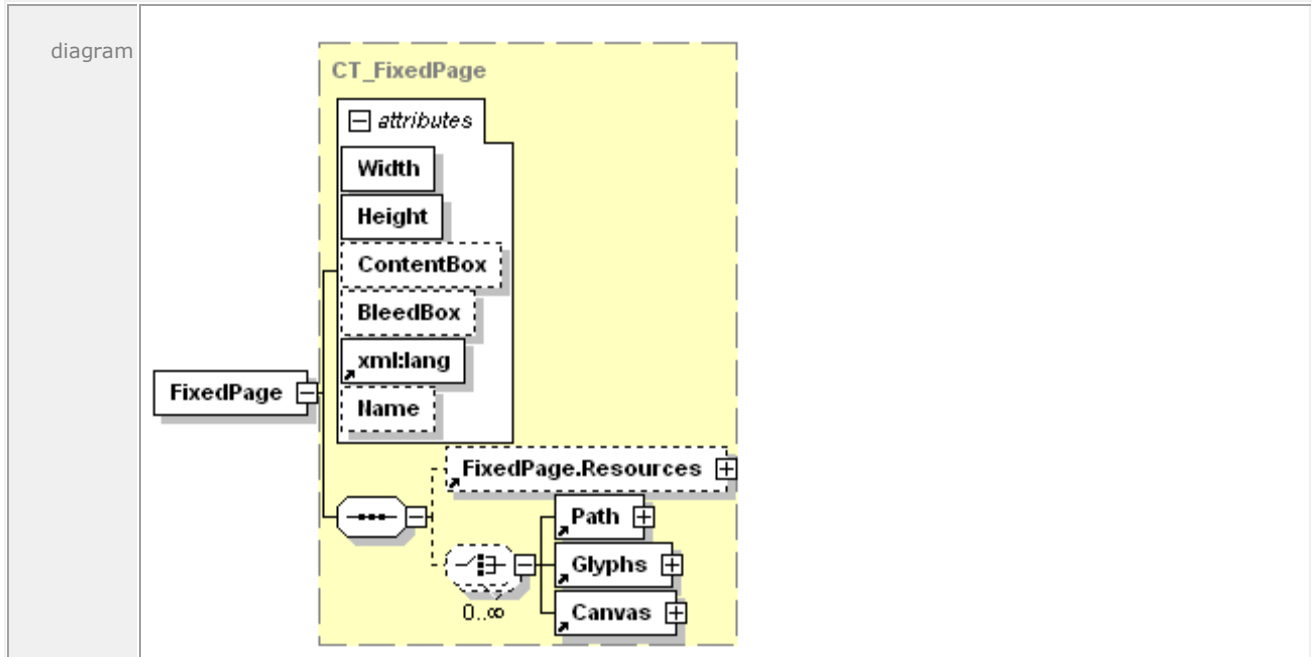
8 element **FixedDocumentSequence**

diagram	
annotation	Specifies a sequence of fixed documents.

9 For more information, see §10.1.

10 **19.16 FixedPage**

11 element **FixedPage**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Width	<u>ST_GEOne</u>	required			Width of the page, expressed as a real number in units of the effective coordinate space.
	Height	<u>ST_GEOne</u>	required			Height of the page, expressed as a real number in units of the effective coordinate space.
	ContentBox	<u>ST_ContentBox</u>				Specifies the area of the page containing imageable content that is to be fit within the imageable area when printing or viewing. Contains a list of four coordinate values (ContentOriginX, ContentOriginY, ContentWidth, ContentHeight), expressed as comma-separated real numbers. Specifying a value is RECOMMENDED [S3.1]. If omitted, the default value is (0,0,Width,Height).
	BleedBox	<u>ST_BleedBox</u>				Specifies the area including crop marks that extends outside of the physical page. Contains a list of four coordinate values (BleedOriginX, BleedOriginY, BleedWidth, BleedHeight), expressed as comma-separated real numbers. If omitted, the default value is (0,0,Width,Height).
	xml:lang		required			Specifies the default language used for the current element and for any child or descendant elements. The language is specified according to RFC 3066.
	Name	<u>ST_Name</u>				Contains a string value that identifies the current

						element as a named, addressable point in the document for the purpose of hyperlinking.
annotation	Contains markup that describes the rendering of a single page of content.					

1 For more information, see §10.3.

2 **19.17 FixedPage.Resources**

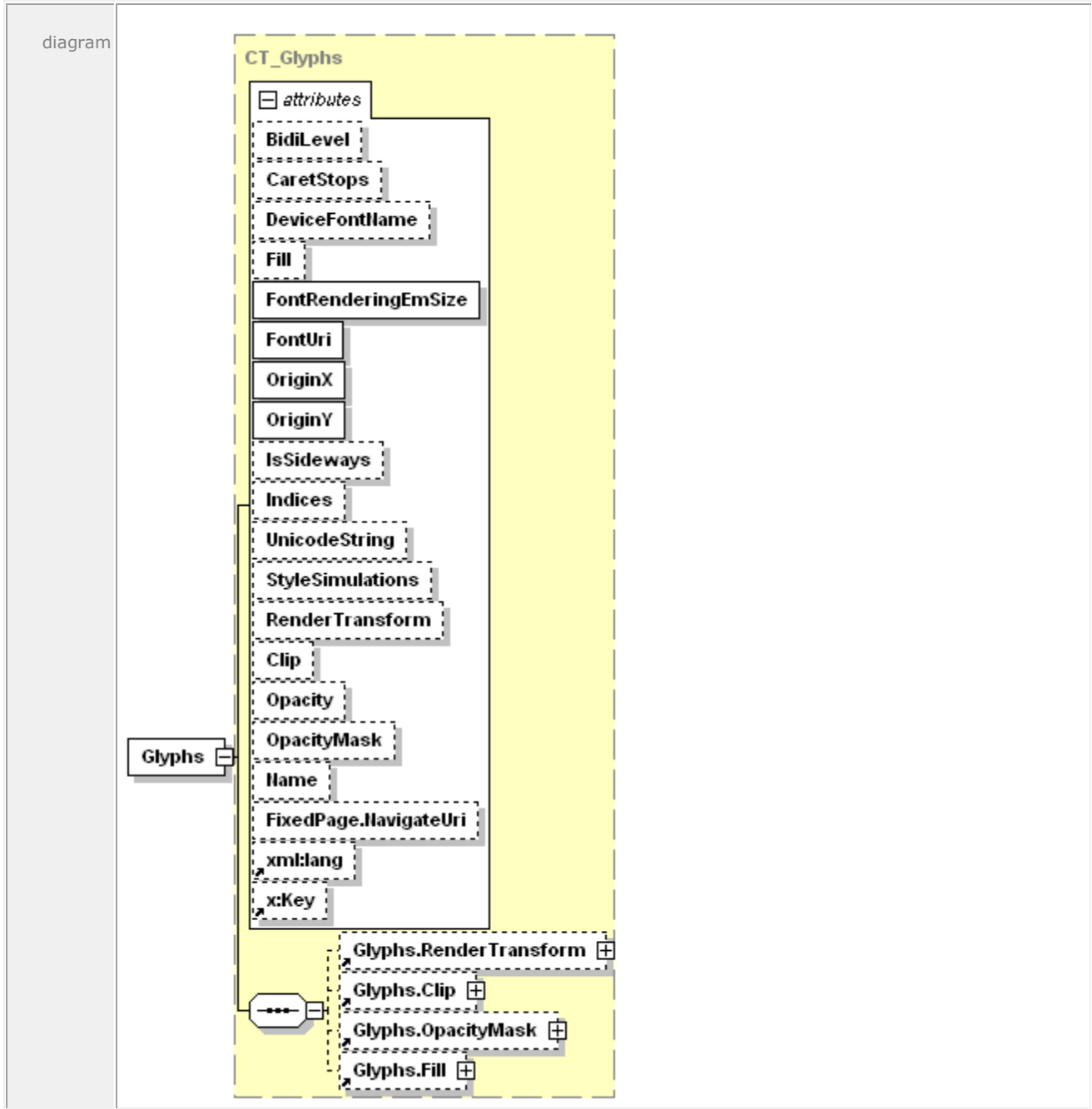
3 element **FixedPage.Resources**

diagram	<pre> classDiagram class FixedPageResources["FixedPage.Resources"] class ResourceDictionary FixedPageResources ..> ResourceDictionary </pre>					
annotation	Contains the resource dictionary for the <FixedPage> element.					

4 For more information, see §14.2.

5 **19.18 Glyphs**

7 element **Glyphs**



attributes	Name	Type	Use	Default	Fixed	Annotation
	BidiLevel			0		Specifies the Unicode algorithm bidirectional nesting level. Even values imply left-to-right layout, odd values imply right-to-left layout. Right-to-left layout places the run origin at the right side of the first glyph, with positive advance widths (representing

					advances to the left) placing subsequent glyphs to the left of the previous glyph. Valid values range from 0 to 61, inclusive.
CaretStops	<u>ST_CaretStops</u>				Identifies the positions within the sequence of Unicode characters at which a text-selection tool can place a text-editing caret. Potential caret-stop positions are identified by their indices into the UTF-16 code units represented by the UnicodeString attribute value. When this attribute is missing, the text in the UnicodeString attribute value MUST be interpreted as having a caret stop between every Unicode UTF-16 code unit and at the beginning and end of the text [M5.1]. The value SHOULD indicate that the caret cannot stop in front of most combining marks or in front of the second UTF-16 code unit of UTF-16 surrogate pairs [S5.1].
DeviceFontName	<u>ST_UnicodeString</u>				Uniquely identifies a specific device font. The identifier is typically defined by a hardware vendor or font vendor.
Fill	<u>ST_RscRefColor</u>				Describes the brush used to fill the shape of the rendered glyphs.
FontRenderingEmSize	<u>ST_GEZero</u>	required			Specifies the font size in drawing surface units, expressed as a float in units of the effective coordinate space. A value of 0 results in no visible text.
FontUri	xs:anyURI	required			The URI of the physical font from which all glyphs in the run are drawn. The URI MUST reference a font contained in the package [M2.1]. If the physical font referenced is a TrueType Collection (containing multiple font faces), the fragment portion of the URI is a 0-based index indicating which font face of the TrueType Collection should be used.

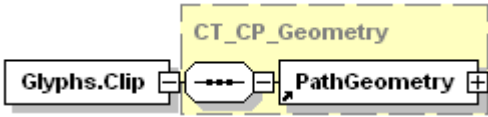
OriginX	<u>ST_Double</u>	required			Specifies the x coordinate of the first glyph in the run, in units of the effective coordinate space. The glyph is placed so that the leading edge of its advance vector and its baseline intersect with the point defined by the OriginX and OriginY attributes.
OriginY	<u>ST_Double</u>	required			Specifies the y coordinate of the first glyph in the run, in units of the effective coordinate space. The glyph is placed so that the leading edge of its advance vector and its baseline intersect with the point defined by the OriginX and OriginY attributes.
IsSideways	<u>ST_Boolean</u>		false		Indicates that a glyph is turned on its side, with the origin being defined as the top center of the unturned glyph.
Indices	<u>ST_Indices</u>				Specifies a series of glyph indices and their attributes used for rendering the glyph run. If the UnicodeString attribute specifies an empty string (" or "{}") and the Indices attribute is not specified or is also empty, a consumer MUST generate an error [M5.2].
UnicodeString	<u>ST_UnicodeString</u>				Contains the string of text rendered by the <Glyphs> element. The text is specified as Unicode code points.
StyleSimulations	<u>ST_StyleSimulations</u>		None		Specifies a style simulation. Valid values are None, ItalicSimulation, BoldSimulation, and BoldItalicSimulation.
RenderTransform	<u>ST_RscRefMatrix</u>				Establishes a new coordinate frame for the glyph run specified by the <Glyphs> element. The render transform affects clip, opacity mask, fill, x origin, y origin, the actual shape of individual glyphs, and the advance widths. The render transform also affects the font size and values specified in the

					Indices attribute.
Clip	ST_RscRefAbbrGeomF				Limits the rendered region of the element. Only portions of the <Glyphs> element that fall within the clip region (even partially clipped characters) produce marks on the page.
Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the glyph element. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
OpacityMask	ST_RscRef				Specifies a mask of alpha values that is applied to the glyphs in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
Name	ST_Name				Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of hyperlinking.
FixedPage.NavigateUri	xs:anyURI				Associates a hyperlink URI with the element. May be a relative reference or a URI that addresses a resource that is internal to or external to the package.
xml:lang					Specifies the default language used for the current element. The language is specified according to RFC 3066.
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M5.3].
annotation	Represents a run of text from a single font.				

1 For more information, see §12.1, §9.1.7, and §12.1.3.

19.19 Glyphs.Clip

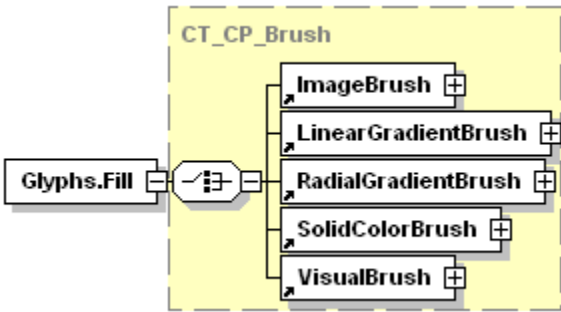
element **Glyphs.Clip**

diagram	
annotation	Limits the rendered region of the element. Only portions of the <Glyphs> element that fall within the clip region (even partially clipped characters) produce marks on the page.

For more information, see §14.3.

19.20 Glyphs.Fill

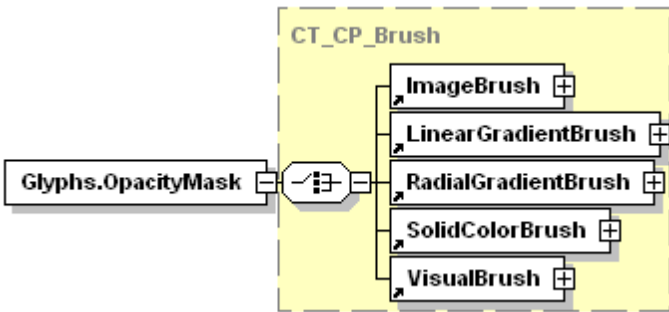
element **Glyphs.Fill**

diagram	
annotation	Describes the brush used to fill the shape of the rendered glyphs.

For more information, see §12.2.

19.21 Glyphs.OpacityMask

element **Glyphs.OpacityMask**

diagram	
annotation	Specifies a mask of alpha values that is applied to the glyphs in the same fashion as the Opacity attribute, but

allowing different alpha values for different areas of the element.

1 For more information, see §14.5.3.

2 19.22 Glyphs.RenderTransform

3 element **Glyphs.RenderTransform**

diagram	
annotation	<p>Establishes a new coordinate frame for the glyph run specified by the <Glyphs> element. The render transform affects clip, opacity mask, fill, x origin, y origin, the actual shape of individual glyphs, and the advance widths. The render transform also affects the font size and values specified in the Indices attribute.</p>

4 For more information, see §14.4.

5 19.23 GradientStop

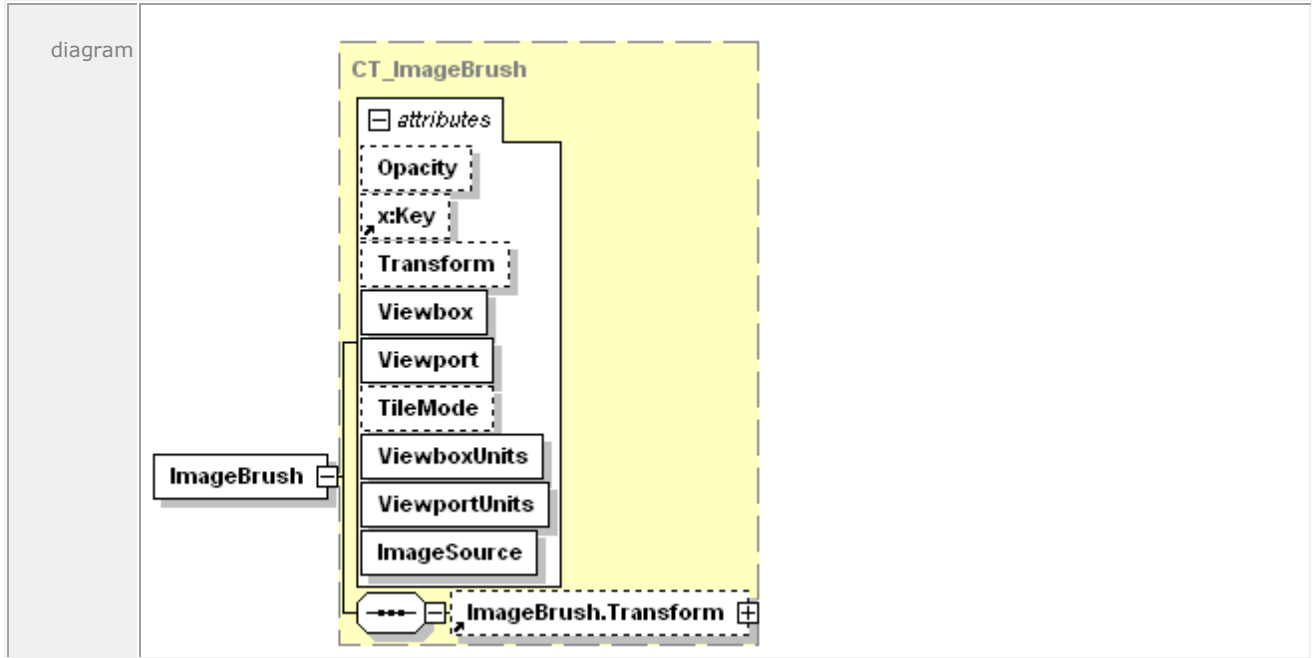
6 element **GradientStop**

diagram																			
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Color</td> <td><u>ST_Color</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the gradient stop color. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.</td> </tr> <tr> <td>Offset</td> <td><u>ST_Double</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the gradient offset. The offset indicates a point along the progression of the gradient at which a color is specified. Colors between gradient offsets in the progression are interpolated.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Color	<u>ST_Color</u>	required			Specifies the gradient stop color. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.	Offset	<u>ST_Double</u>	required			Specifies the gradient offset. The offset indicates a point along the progression of the gradient at which a color is specified. Colors between gradient offsets in the progression are interpolated.
Name	Type	Use	Default	Fixed	Annotation														
Color	<u>ST_Color</u>	required			Specifies the gradient stop color. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.														
Offset	<u>ST_Double</u>	required			Specifies the gradient offset. The offset indicates a point along the progression of the gradient at which a color is specified. Colors between gradient offsets in the progression are interpolated.														
annotation	<p>Indicates a location and range of color progression for rendering a gradient.</p>																		

7 For more information, see §13.7.

1 **19.24 ImageBrush**

2 element **ImageBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.2].
	Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.
	Viewbox	ST_ViewBox	required			Specifies the position and dimensions of the brush's source content. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The dimensions specified are relative to the image's physical dimensions expressed in

					units of 1/96". The corners of the viewbox are mapped to the corners of the viewport, thereby providing the default clipping and transform for the brush's source content.
Viewport	ST_ViewBox	required			Specifies the region in the containing coordinate space of the prime brush tile that is (possibly repeatedly) applied to fill the region to which the brush is applied. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The alignment of the brush pattern is controlled by adjusting the x and y values.
TileMode	ST_TileMode		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
ImageSource	ST_UriCtxBmp	required			Specifies the URI of an image resource or a combination of the URI of an image resource a color profile resource. See the Color clause for important details. The URI MUST refer to parts in the package [M2.1].
annotation	Fills a region with an image.				

1 For more information, see §13.2.

2 19.25 ImageBrush.Transform

3 element **ImageBrush.Transform**

diagram	
annotation	Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.

1 For more information, see §14.4.

2 **19.26 Intent**

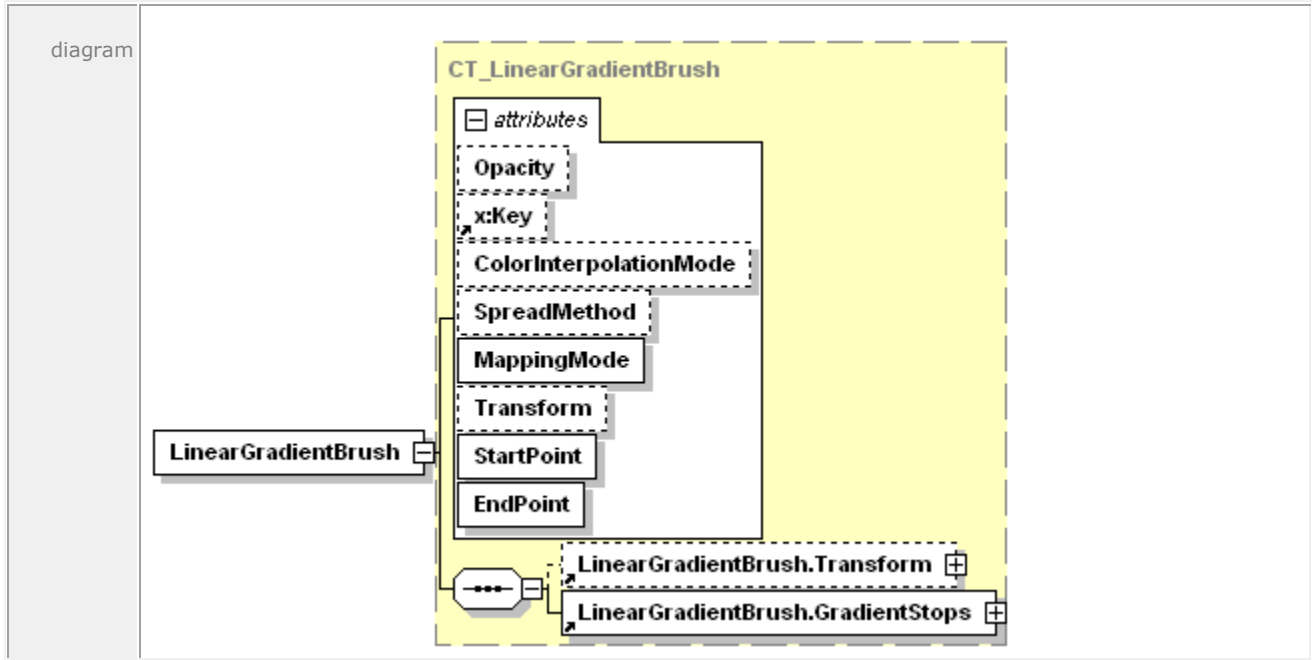
3 element **SignatureDefinitionType/Intent**

diagram	
annotation	A string that represents the intent to which the signing party agrees when signing the document.

4 For more information, see §17.2.2.4.

5 **19.27 LinearGradientBrush**

6 element **LinearGradientBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the linear gradient. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the

					current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.5].
ColorInterpolationMode	<u>ST_ClrIntMode</u>		SRgbLinear Interpolation		Specifies the gamma function for color interpolation. The gamma adjustment should not be applied to the alpha component, if specified. Valid values are SRgbLinearInterpolation and ScRgbLinearInterpolation.
SpreadMethod	<u>ST_Spread Method</u>		Pad		Describes how the brush should fill the content area outside of the primary, initial gradient area. Valid values are Pad, Reflect and Repeat.
MappingMode	<u>ST_Mapping Mode</u>	required		Absolute	Specifies that the start point and end point are defined in the effective coordinate space (includes the Transform attribute of the brush).
Transform	<u>ST_RscRef Matrix</u>				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property on a brush is concatenated with the current effective render transform to yield an effective render transform local to the brush. The start point and end point are transformed using the local effective render transform.
StartPoint	<u>ST_Point</u>	required			Specifies the starting point of the linear gradient.
EndPoint	<u>ST_Point</u>	required			Specifies the end point of the linear gradient. The linear gradient brush interpolates the colors from the start point to the end point, where the start point represents an offset of 0, and the EndPoint represents an offset of 1. The Offset attribute value specified in a GradientStop element relates to the 0 and 1 offsets defined by the start point and end point.

annotation	Fills a region with a linear gradient.
------------	--

1 For more information, see §13.5 and §15.

2 **19.28 LinearGradientBrush.GradientStops**

3 element **LinearGradientBrush.GradientStops**

diagram	
annotation	Holds a sequence of GradientStop elements.

4 For more information, see §13.5.2.

5 **19.29 LinearGradientBrush.Transform**

6 element **LinearGradientBrush.Transform**

diagram	
annotation	Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The start point and end point are transformed using the local effective render transform.

7 For more information, see §14.4.8.

8 **19.30 LinkTarget**

9 element **LinkTarget**

diagram													
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Name</td> <td><u>ST_Name</u></td> <td>required</td> <td></td> <td></td> <td>Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Name	<u>ST_Name</u>	required			Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of
Name	Type	Use	Default	Fixed	Annotation								
Name	<u>ST_Name</u>	required			Contains a string value that identifies the current element as a named, addressable point in the document for the purpose of								

						hyperlinking.
annotation	Specifies an addressable point on the page.					

1 For more information, see §10.2.3.

2 19.31 ListItemStructure

3 element **ListItemStructure**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Marker	<u>ST_NameUnique</u>	optional			The named element that represents the marker for this list items, such as a bullet, number, or image.
annotation	Describes a single structural block. These structural blocks are grouped together in a list.					

4 For more information, see §16.1.2.11.

5 19.32 ListStructure

6 element **ListStructure**

diagram						
annotation	Contains a collection of items that are group together in a list.					

7 For more information, see §16.1.2.10.

1 **19.33 MatrixTransform**

2 element **MatrixTransform**

diagram	<p>The diagram shows a box labeled 'MatrixTransform' connected to a larger box labeled 'CT_MatrixTran'. Inside 'CT_MatrixTran', there is a sub-section 'attributes' containing two items: 'Matrix' and 'x:Key'.</p>					
attributes	Name	Type	Use	Default	Fixed	Annotation
	Matrix	<u>ST_Matrix</u>	required			Specifies the matrix structure that defines the transformation.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M7.11].
annotation	Creates an arbitrary affine matrix transformation that manipulates objects or coordinate systems in a two-dimensional plane.					

3 For more information, see §14.4.1.

4 **19.34 NamedElement**

5 element **NamedElement**

diagram	<p>The diagram shows a box labeled 'NamedElement' connected to a larger box labeled 'CT_NamedElement'. Inside 'CT_NamedElement', there is a sub-section 'attributes' containing one item: 'NameReference'.</p>					
attributes	Name	Type	Use	Default	Fixed	Annotation
	NameReference	<u>ST_Name</u>	required			Identifies the named element in the FixedPage part markup that is referenced by the document structure markup.
annotation	All document structure is related to the fixed page markup using this element. The <NamedElement> points to a single markup element contained in the fixed page markup.					

6 For more information, see §16.1.2.13.

1 **19.35 OutlineEntry**

2 element **OutlineEntry**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	OutlineLevel	<u>ST_IntGEOne</u>	optional	1		A description of the level where the outline entry exists in the hierarchy. A value of 1 is the root.
	OutlineTarget	xs:anyURI	required			The URI to which the outline entry is linked. This can be a URI to a named element within the document or an external URI, such as a website. It can be used as a hyperlink destination.
	Description	xs:string	required			The friendly text associated with this outline entry.
	xml:lang		optional			This attribute specifies the default language used for any child element contained within the current element or nested child elements. The language is specified according to IETF RFC 3066.
annotation	Represents an index to a specific location in the document.					

3 For more information, see §16.1.1.4.

4 **19.36 PageContent**

5 element **PageContent**

<p>diagram</p>																														
<p>attributes</p>	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Source</td> <td>xs:anyURI</td> <td>required</td> <td></td> <td></td> <td>Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].</td> </tr> <tr> <td>Width</td> <td>ST_GEOne</td> <td></td> <td></td> <td></td> <td>Typical width of pages contained in the page content.</td> </tr> <tr> <td>Height</td> <td>ST_GEOne</td> <td></td> <td></td> <td></td> <td>Typical height of pages contained in the page content.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Source	xs:anyURI	required			Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].	Width	ST_GEOne				Typical width of pages contained in the page content.	Height	ST_GEOne				Typical height of pages contained in the page content.					
Name	Type	Use	Default	Fixed	Annotation																									
Source	xs:anyURI	required			Specifies a URI that refers to the page content, held in a distinct part within the package. The content identified MUST be a FixedPage part within the XPS Document [M3.5].																									
Width	ST_GEOne				Typical width of pages contained in the page content.																									
Height	ST_GEOne				Typical height of pages contained in the page content.																									
<p>annotation</p>	<p>Defines a reference from a fixed document to a part that contains a <FixedPage> element.</p>																													

1 For more information, see §10.2.1.

2 19.37 PageContent.LinkTargets

3 element **PageContent.LinkTargets**

<p>diagram</p>						
<p>annotation</p>	<p>Contains a collection of <LinkTarget> elements, each of which is addressable via hyperlink.</p>					

4 For more information, see §10.2.2.

5 19.38 ParagraphStructure

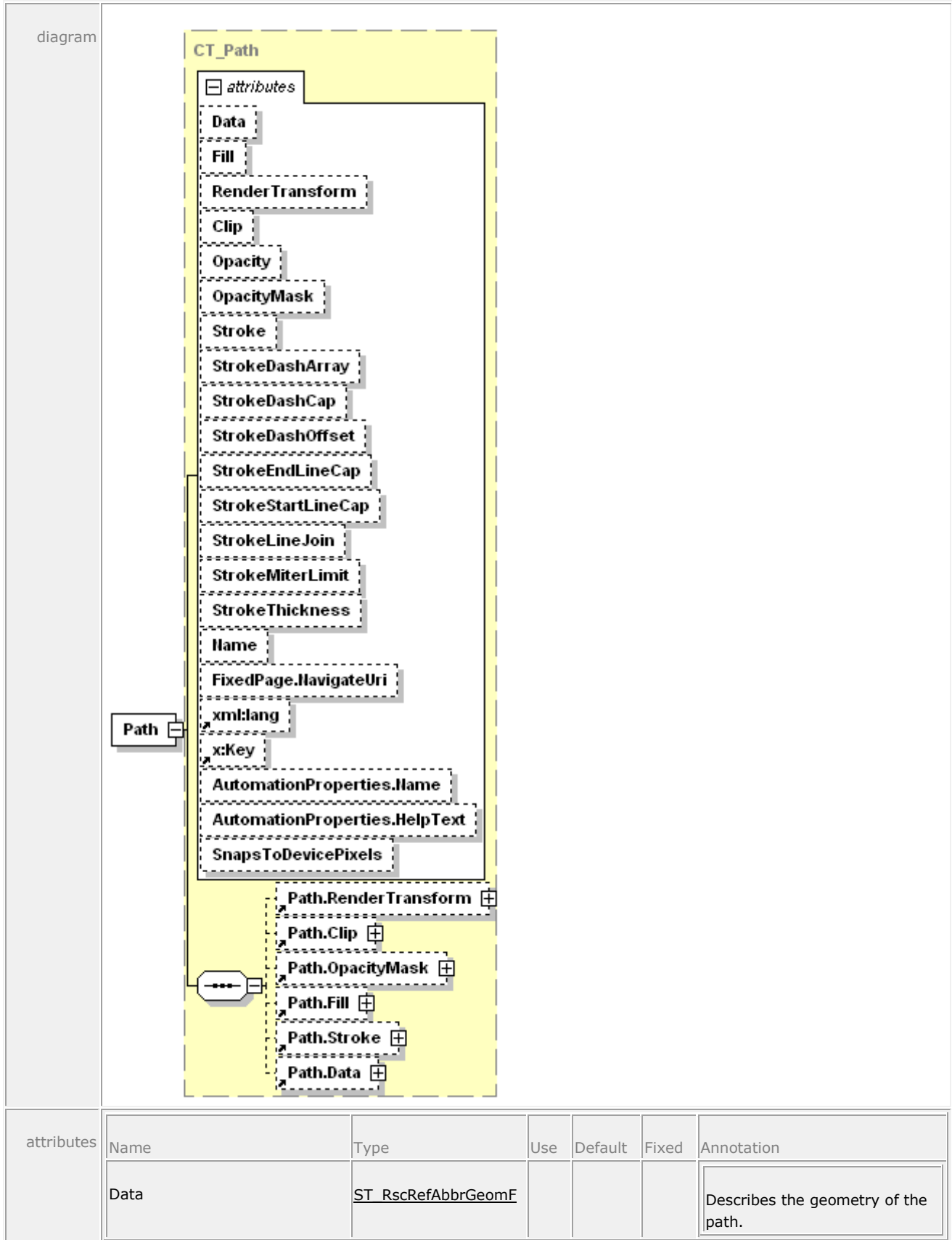
6 element **ParagraphStructure**

<p>diagram</p>	<pre> classDiagram class ParagraphStructure class CT_Paragraph class NamedElement ParagraphStructure -- "0..∞" CT_Paragraph CT_Paragraph -- NamedElement </pre>
<p>annotation</p>	<p>Contains the named elements that constitute a single paragraph.</p>

1 For more information, see §16.1.2.5.

2 **19.39 Path**

3 element **Path**



Fill	ST_RscRefColor				Describes the brush used to paint the geometry specified by the Data property of the path.
RenderTransform	ST_RscRefMatrix				Establishes a new coordinate frame for all attributes of the path and for all child elements of the path, such as the geometry defined by the <Path.Data> property element.
Clip	ST_RscRefAbbrGeomF				Limits the rendered region of the element.
Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the path element. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
OpacityMask	ST_RscRef				Specifies a mask of alpha values that is applied to the path in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.
Stroke	ST_RscRefColor				Specifies the brush used to draw the stroke.
StrokeDashArray	ST_EvenArrayPos				Specifies the length of dashes and gaps of the outline stroke. These values are specified as multiples of the stroke thickness as a space-separated list with an even number of non-negative values. When a stroke is drawn, the dashes and gaps specified by these values are repeated to cover the length of the stroke. If this attribute is omitted, the stroke is drawn solid, without any gaps.
StrokeDashCap	ST_DashCap		Flat		Specifies how the ends of each dash are drawn. Valid values are Flat, Round, Square, and

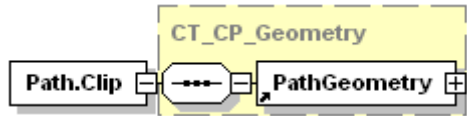
					Triangle.
StrokeDashOffset	<u>ST_Double</u>		0.0		Adjusts the start point for repeating the dash array pattern. If this value is omitted, the dash array aligns with the origin of the stroke. Values are specified as multiples of the stroke thickness.
StrokeEndLineCap	<u>ST_LineCap</u>		Flat		Defines the shape of the end of the last dash in a stroke. Valid values are Flat, Square, Round, and Triangle.
StrokeStartLineCap	<u>ST_LineCap</u>		Flat		Defines the shape of the beginning of the first dash in a stroke. Valid values are Flat, Square, Round, and Triangle.
StrokeLineJoin	<u>ST_LineJoin</u>		Miter		Specifies how a stroke is drawn at a corner of a path. Valid values are Miter, Bevel, and Round. If Miter is selected, the value of StrokeMiterLimit is used in drawing the stroke.
StrokeMiterLimit	<u>ST_GEOne</u>		10.0		The ratio between the maximum miter length and half of the stroke thickness. This value is significant only if the StrokeLineJoin attribute specifies Miter.
StrokeThickness	<u>ST_GEZero</u>		1.0		Specifies the thickness of a stroke, in units of the effective coordinate space (includes the path's render transform). The stroke is drawn on top of the boundary of the geometry specified by the <Path> element's Data property. Half of the StrokeThickness extends outside of the geometry specified by the Data property and the other half extends inside of the geometry.
Name	<u>ST_Name</u>				Contains a string value that identifies the current element

					as a named, addressable point in the document for the purpose of hyperlinking.
	FixedPage.NavigateUri	xs:anyURI			Associates a hyperlink URI with the element. Can be a relative reference or a URI that addresses a resource that is internal to or external to the package.
	xml:lang				Specifies the default language used for the current element and for any child or descendant elements. The language is specified according to RFC 3066.
	x:Key				Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.1].
	AutomationProperties.Name	xs:string			A brief description of the <Path> for accessibility purposes, particularly if filled with an <ImageBrush>.
	AutomationProperties.HelpText	xs:string			A detailed description of the <Path> for accessibility purposes, particularly if filled with an <ImageBrush>.
	SnapsToDevicePixels	<u>ST_Boolean</u>			On Anti-aliasing consumers controls if control points snap to the nearest device pixels. Valid values are 'false' and 'true'. Consumers MAY ignore this attribute [O4.1].
annotation	Defines a single graphical effect to be rendered to the page. It paints a geometry with a brush and draws a stroke around it.				

- 1 For more information, see §11.1 and §11.2.3.

1 19.40 Path.Clip

2 element **Path.Clip**

diagram	
annotation	Limits the rendered region of the element.

3 For more information, see §14.3.

4 19.41 Path.Data

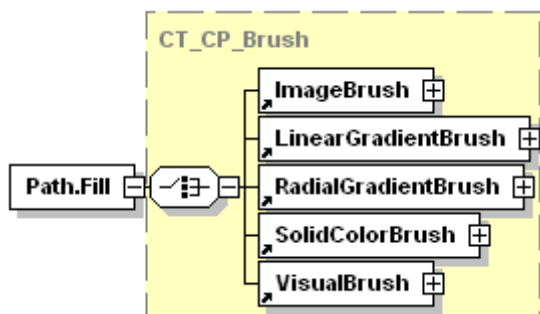
5 element **Path.Data**

diagram	
annotation	Describes the geometry of the path.

6 For more information, see §11.1.1.

7 19.42 Path.Fill

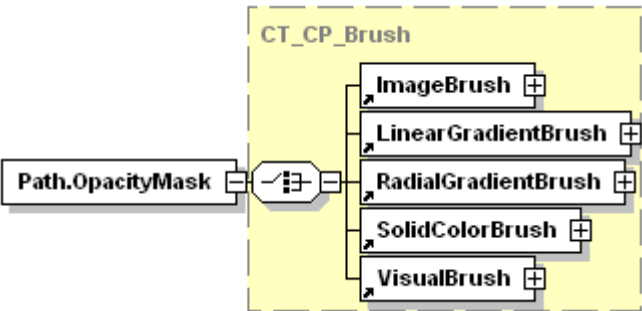
8 element **Path.Fill**

diagram	
annotation	Describes the brush used to paint the geometry specified by the Data property of the path.

9 For more information, see §11.1.2.

10 19.43 Path.OpacityMask

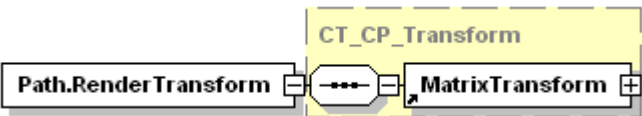
11 element **Path.OpacityMask**

<p>diagram</p>	
<p>annotation</p>	<p>Specifies the mask of alpha values that is applied to the path in the same fashion as the Opacity attribute, but allowing different alpha values for different areas of the element.</p>

1 For more information, see §14.5.2.

2 19.44 Path.RenderTransform

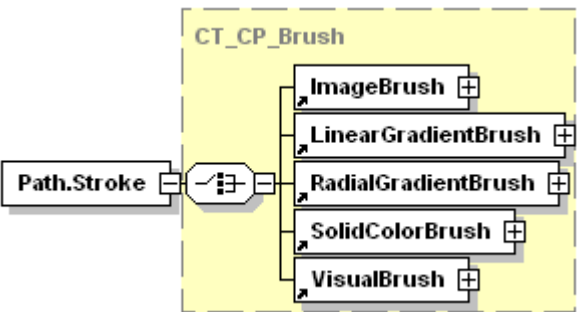
3 element **Path.RenderTransform**

<p>diagram</p>	
<p>annotation</p>	<p>Establishes a new coordinate frame for all attributes of the path and for all child elements of the path, such as the geometry defined by the <Path.Data> property element.</p>

4 For more information, see §14.4.

5 19.45 Path.Stroke

6 element **Path.Stroke**

<p>diagram</p>	
<p>annotation</p>	<p>Specifies the brush used to draw the stroke.</p>

7

1 For more information, see §11.1.3.

2 **19.46 PathFigure**

3 element **PathFigure**

diagram

attributes

Name	Type	Use	Default	Fixed	Annotation
IsClosed	ST_Boolean		false		Specifies whether the path is closed. If set to true, the stroke is drawn "closed," that is, the last point in the last segment of the path figure is connected with the point specified in the StartPoint attribute, otherwise the stroke is drawn "open," and the last point is not connected to the start point. Only applicable if the path figure is used in a <Path> element that specifies a stroke.
StartPoint	ST_Point	required			Specifies the starting point for the first segment of the path figure.
IsFilled	ST_Boolean		true		Specifies whether the path figure is used in computing the area of the containing path geometry. Can be true or false. When set to false, the path figure is considered only for stroking.

annotation

Specifies a set of one or more segment elements defining a closed region.

4 For more information, see §11.2.2.1.

5 **19.47 PathGeometry**

6 element **PathGeometry**

<p>diagram</p>																																				
<p>attributes</p>	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Figures</td> <td><u>ST_AbbrGeom</u></td> <td></td> <td></td> <td></td> <td>Describes the geometry of the path.</td> </tr> <tr> <td>FillRule</td> <td><u>ST_FillRule</u></td> <td></td> <td>EvenOdd</td> <td></td> <td>Specifies how the intersecting areas of geometric shapes are combined to form a region. Valid values are EvenOdd and NonZero.</td> </tr> <tr> <td>Transform</td> <td><u>ST_RscRefMatrix</u></td> <td></td> <td></td> <td></td> <td>Specifies the local matrix transformation that is applied to all child and descendant elements of the path geometry before it is used for filling, clipping, or stroking.</td> </tr> <tr> <td>x:Key</td> <td></td> <td></td> <td></td> <td></td> <td>Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.2].</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Figures	<u>ST_AbbrGeom</u>				Describes the geometry of the path.	FillRule	<u>ST_FillRule</u>		EvenOdd		Specifies how the intersecting areas of geometric shapes are combined to form a region. Valid values are EvenOdd and NonZero.	Transform	<u>ST_RscRefMatrix</u>				Specifies the local matrix transformation that is applied to all child and descendant elements of the path geometry before it is used for filling, clipping, or stroking.	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.2].					
Name	Type	Use	Default	Fixed	Annotation																															
Figures	<u>ST_AbbrGeom</u>				Describes the geometry of the path.																															
FillRule	<u>ST_FillRule</u>		EvenOdd		Specifies how the intersecting areas of geometric shapes are combined to form a region. Valid values are EvenOdd and NonZero.																															
Transform	<u>ST_RscRefMatrix</u>				Specifies the local matrix transformation that is applied to all child and descendant elements of the path geometry before it is used for filling, clipping, or stroking.																															
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M4.2].																															
<p>annotation</p>	<p>Contains a set of <PathFigure> elements.</p>																																			

For more information, see §11.2.1.1.

19.48 PathGeometry.Transform

element **PathGeometry.Transform**

<p>diagram</p>						
<p>annotation</p>	<p>Specifies the local matrix transformation that is applied to all child and descendant elements of the path</p>					

geometry before it is used for filling, clipping, or stroking.

1 For more information, see §14.4.5.

2 19.49 PolyBezierSegment

3 element **PolyBezierSegment**

diagram																								
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Points</td> <td><u>ST_Points</u></td> <td>required</td> <td></td> <td></td> <td>Specifies control points for multiple Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.</td> </tr> <tr> <td>IsStroked</td> <td><u>ST_Boolean</u></td> <td></td> <td>true</td> <td></td> <td>Specifies whether the stroke for this segment of the path is drawn. Can be true or false.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Points	<u>ST_Points</u>	required			Specifies control points for multiple Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.					
Name	Type	Use	Default	Fixed	Annotation																			
Points	<u>ST_Points</u>	required			Specifies control points for multiple Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.																			
IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.																			
annotation	A series of Bézier segments.																							

4

5 19.50 PolyLineSegment

6 element **PolyLineSegment**

diagram																		
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Points</td> <td><u>ST_Points</u></td> <td>required</td> <td></td> <td></td> <td>Specifies a set of coordinates for the multiple segments that define the poly line segment. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Points	<u>ST_Points</u>	required			Specifies a set of coordinates for the multiple segments that define the poly line segment. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.					
Name	Type	Use	Default	Fixed	Annotation													
Points	<u>ST_Points</u>	required			Specifies a set of coordinates for the multiple segments that define the poly line segment. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.													

	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.
1	annotation	Specifies a set of points between which lines are drawn.				

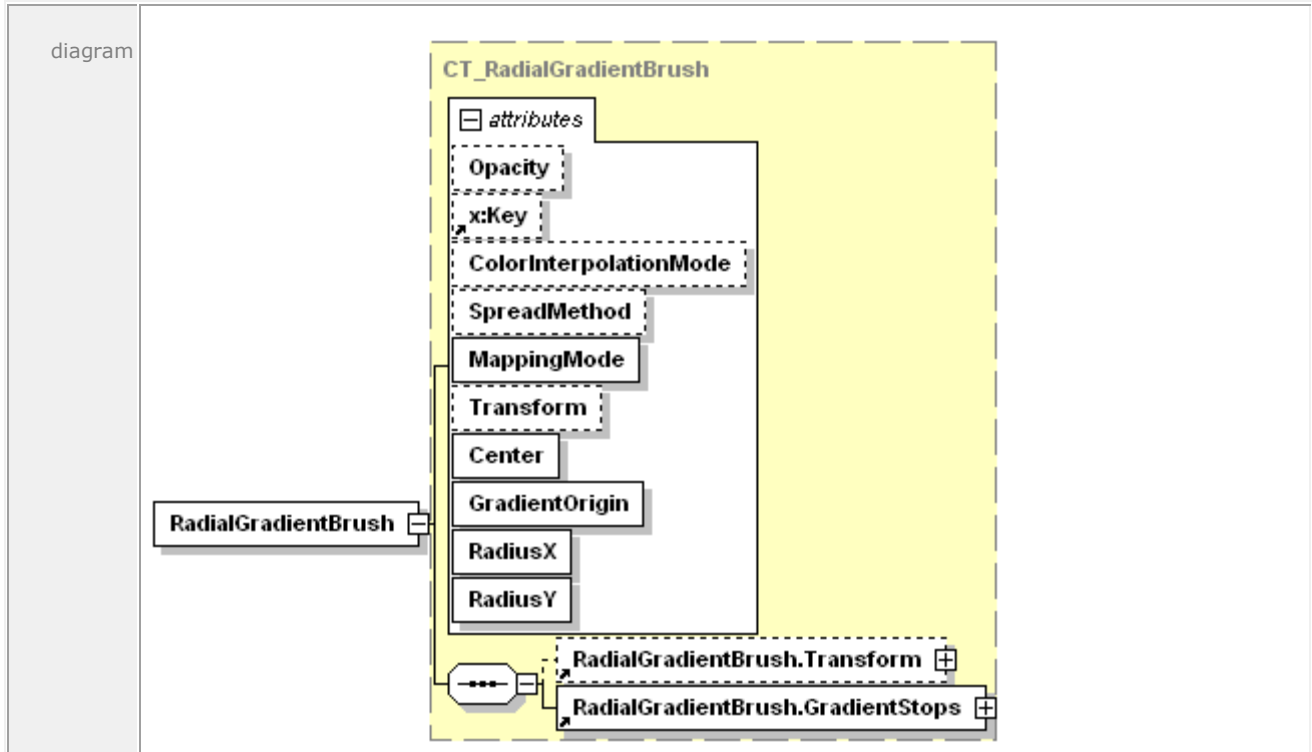
2 **19.51 PolyQuadraticBezierSegment**

3 element **PolyQuadraticBezierSegment**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<u>ST_Points</u>	required			Specifies control points for multiple quadratic Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.
	IsStroked	<u>ST_Boolean</u>		true		Specifies whether the stroke for this segment of the path is drawn. Can be true or false.
4	annotation	A series of quadratic Bézier segments.				

5 **19.52 RadialGradientBrush**

6 element **RadialGradientBrush**




attributes						
Name	Type	Use	Default	Fixed	Annotation	
Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the radial gradient. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.	
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.6].	
ColorInterpolationMode	ST_ClrIntMode		SRgbLinear Interpolation		Specifies the gamma function for color interpolation for sRGB colors. The gamma adjustment should not be applied to the alpha component, if specified. Valid values are SRgbLinearInterpolation and ScRgbLinearInterpolation.	
SpreadMethod	ST_SpreadMethod		Pad		Describes how the brush should fill the content area outside of	

						the primary, initial gradient area. Valid values are Pad, Reflect and Repeat.
	MappingMode	ST_MappingMode	required		Absolute	Specifies that center, x radius, and y radius are defined in the effective coordinate space (includes the Transform attribute of the brush).
	Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The ellipse defined by the center, gradient origin, x radius, and y radius values is transformed using the local effective render transform.
	Center	ST_Point	required			Specifies the center point of the radial gradient (that is, the center of the ellipse). The radial gradient brush interpolates the colors from the gradient origin to the circumference of the ellipse. The circumference is determined by the center and the radii.
	GradientOrigin	ST_Point	required			Specifies the origin point of the radial gradient.
	RadiusX	ST_GEZero	required			Specifies the radius in the x dimension of the ellipse which defines the radial gradient.
	RadiusY	ST_GEZero	required			Specifies the radius in the y dimension of the ellipse which defines the radial gradient.
annotation	Fills a region with a radial gradient.					

- 1 For more information, see §13.6 and §15.

1 **19.53 RadialGradientBrush.GradientStops**

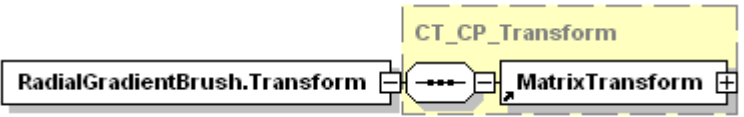
2 element **RadialGradientBrush.GradientStops**

diagram	
annotation	Holds a sequence of <GradientStop> elements.

3 For more information, see §13.6.2.

4 **19.54 RadialGradientBrush.Transform**

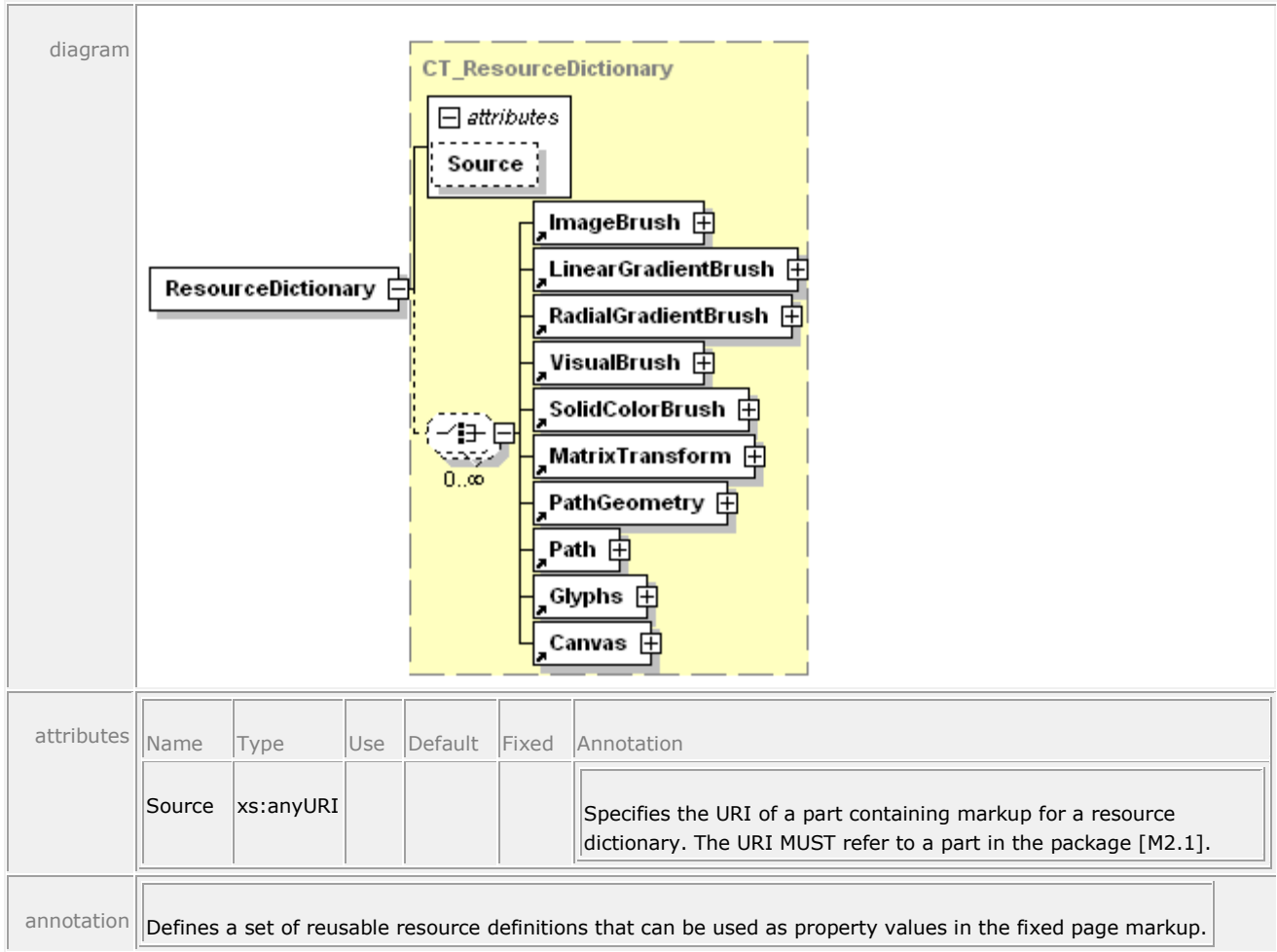
5 element **RadialGradientBrush.Transform**

diagram	
annotation	Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The center, gradient origin, x radius, and y radius are transformed using the local effective render transform.

6 For more information, see §14.4.9.

7 **19.55 ResourceDictionary**

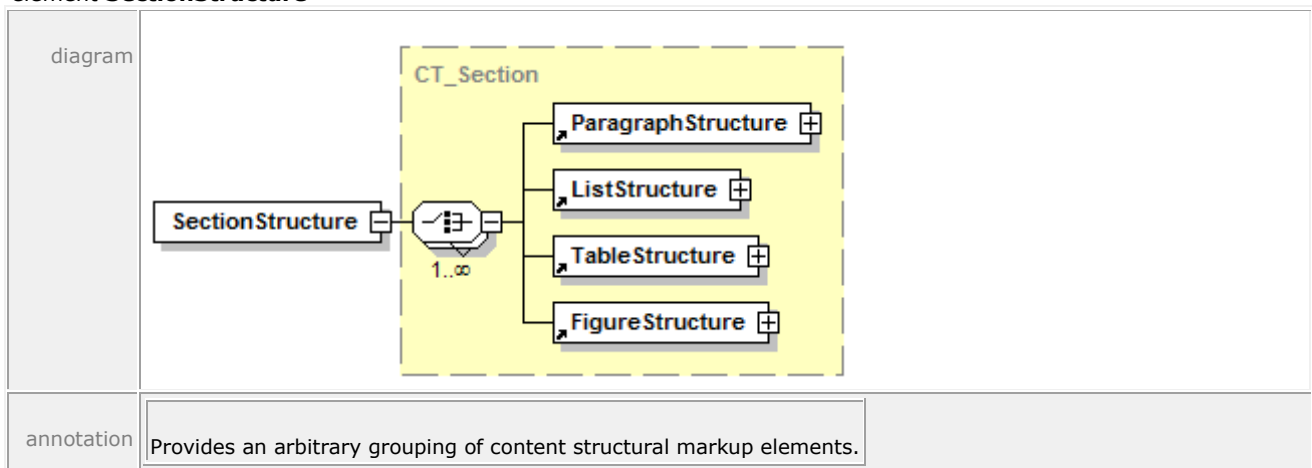
8 element **ResourceDictionary**



1 For more information, see §14.2.

2 19.56 SectionStructure

3 element **SectionStructure**



4 For more information, see §16.1.2.4.

1 **19.57 SignBy**

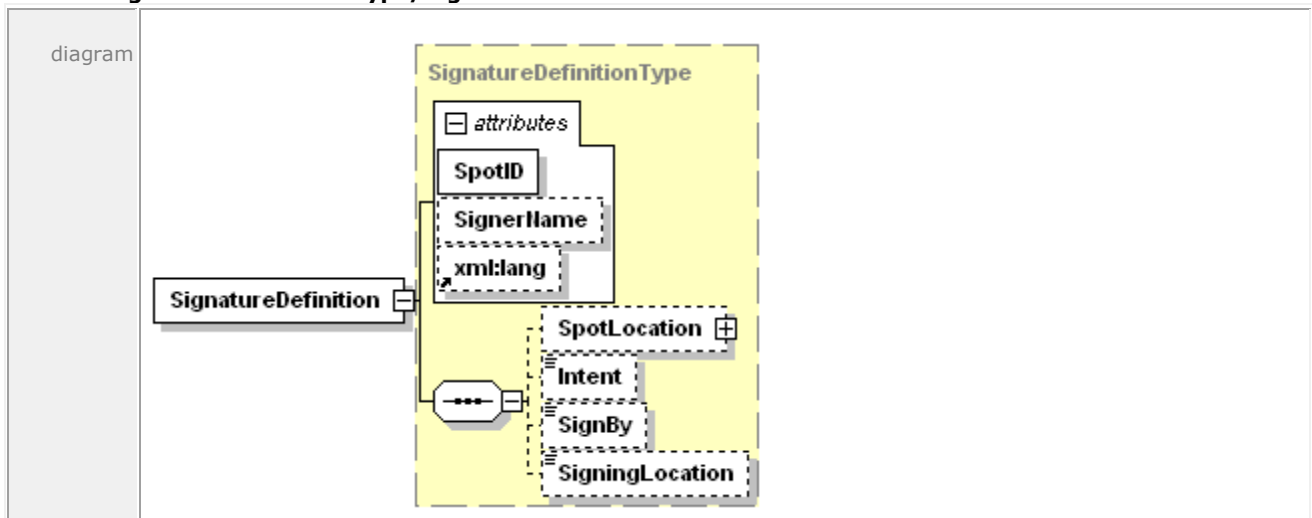
2 element **SignatureDefinitionType/SignBy**

diagram	
annotation	The date and time by which the requested party is to sign the XPS Document.

3 For more information, see §17.2.2.5.

4 **19.58 SignatureDefinition**

5 element **SignatureDefinitionsType/SignatureDefinition**



attributes	Name	Type	Use	Default	Fixed	Annotation
	SpotID	xs:ID	required			
SignerName	xs:string					A string representing the identity of the individual who is requested to sign the XPS Document, or the name of the individual who has signed the XPS Document.
xml:lang						Specifies the language used for the current element and its descendants. The language is specified according to RFC 3066.

annotation	A single signature definition.
------------	--------------------------------

6 For more information, see §17.2.2.2.

1 **19.59 SignatureDefinitions**

2 element **SignatureDefinitions**

diagram	
annotation	The root element for the SignatureDefinitions part.

3 For more information, see §17.2.2.1.

4 **19.60 SigningLocation**

5 element **SignatureDefinitionType/SigningLocation**

diagram	
annotation	The legal location where the document is signed.

6 For more information, see §17.2.2.6.

7 **19.61 SolidColorBrush**

8 element **SolidColorBrush**

diagram																			
attributes	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>Opacity</td> <td><u>ST_ZeroOne</u></td> <td></td> <td>1.0</td> <td></td> <td>Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.</td> </tr> <tr> <td>x:Key</td> <td></td> <td></td> <td></td> <td></td> <td>Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.1].</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.1].
Name	Type	Use	Default	Fixed	Annotation														
Opacity	<u>ST_ZeroOne</u>		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.														
x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.1].														

	Color	<u>ST_Color</u>	required			Specifies the color for filled elements. An sRGB color value specified as a 6-digit hexadecimal number (#RRGGBB) or an extended color.
annotation	Fills defined geometric regions with a solid color.					

1 For more information, see §13.1.

2 19.62 SpotLocation

3 element **SignatureDefinitionType/SpotLocation**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	PageURI	xs:anyURI	required			Specifies the page on which the signature spot should be displayed.
	StartX	xs:double	required			Specifies the x coordinate of the origin point (upper-left corner) on the page where the signature spot should be displayed.
	StartY	xs:double	required			Specifies the y coordinate of the origin point (upper-left corner) on the page where the signature spot should be displayed.
annotation	Specifies where a consumer should place a signature spot.					

4 For more information, see §17.2.2.3.

5 19.63 Story

6 element **Story**

diagram													
attributes	Name	Type	Use	Default	Fixed	Annotation							
annotation	<table border="1"> <tr> <td data-bbox="269 730 410 810">StoryName</td> <td data-bbox="410 730 509 810">xs:string</td> <td data-bbox="509 730 617 810">required</td> <td data-bbox="617 730 712 810"></td> <td data-bbox="712 730 792 810"></td> <td colspan="2" data-bbox="792 730 1437 810">The name used by story fragments to identify they belong to this story.</td> </tr> </table>						StoryName	xs:string	required			The name used by story fragments to identify they belong to this story.	
StoryName	xs:string	required			The name used by story fragments to identify they belong to this story.								

1 For more information, see §16.1.1.5.

2 19.64 StoryBreak

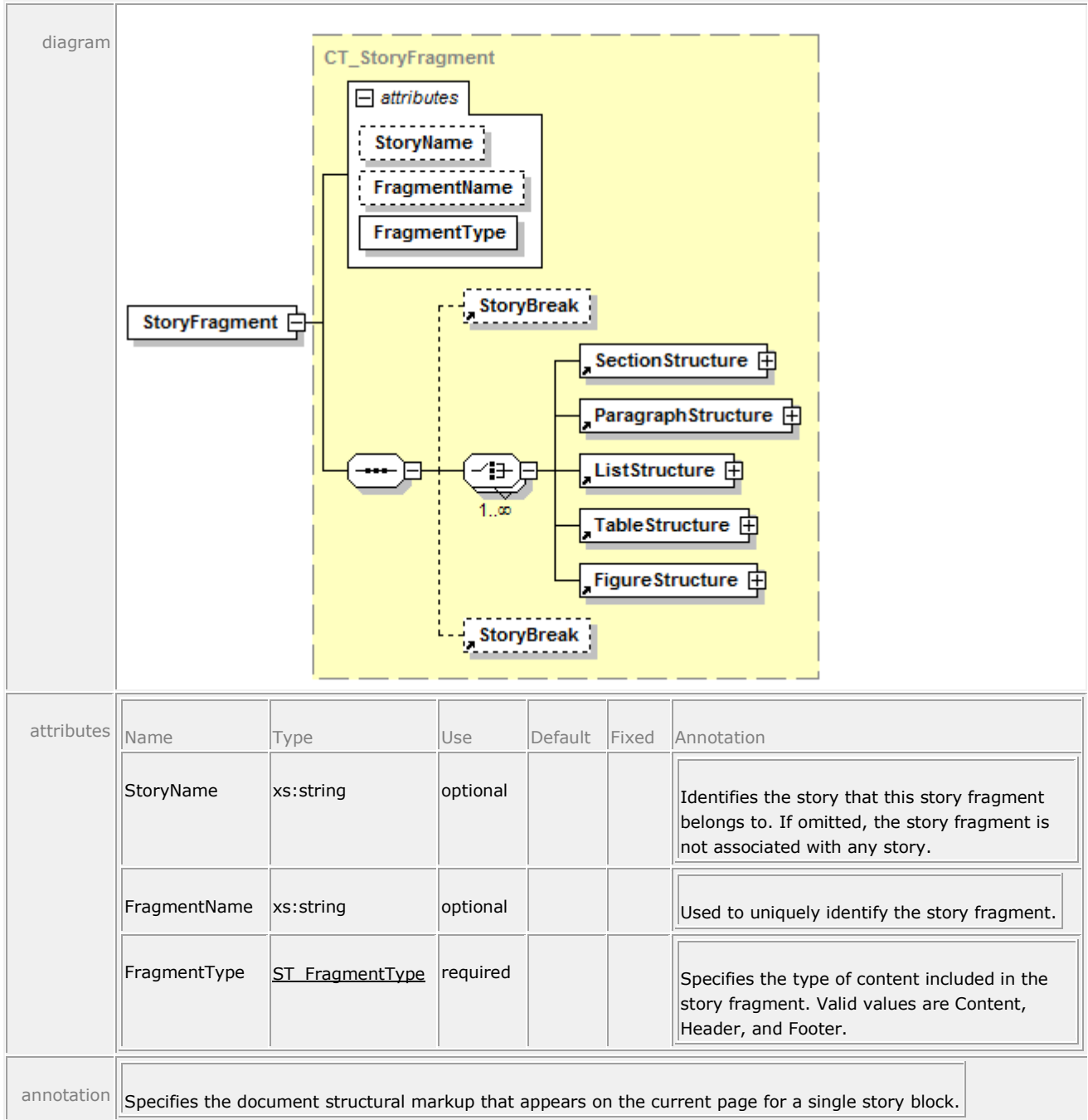
3 element **StoryBreak**

diagram	
annotation	<p>If located at the beginning of a <StoryFragment> definition, indicates that the following markup elements should not be merged with the markup from the previous <StoryFragment>. If located at the end of a <StoryFragment> definition, indicates that the preceding markup elements should not be merged with the subsequent <StoryFragment>.</p>

4 For more information, see §16.1.2.3.

5 19.65 StoryFragment

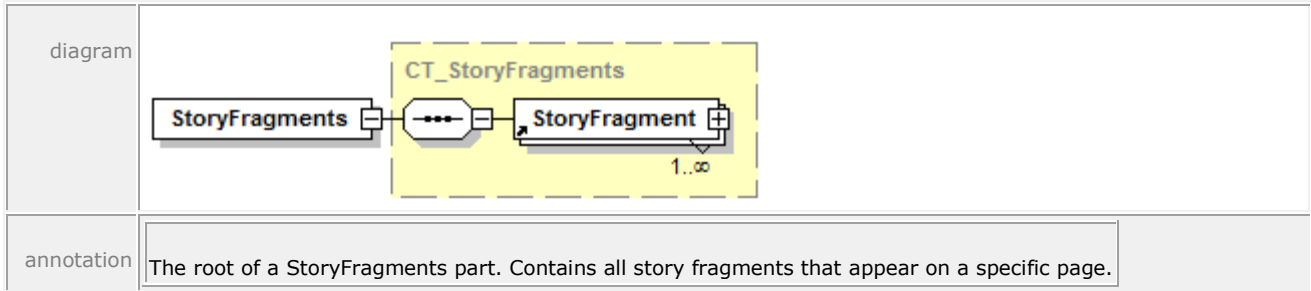
6 element **StoryFragment**



1 For more information, see §16.1.2.2.

2 19.66 StoryFragments

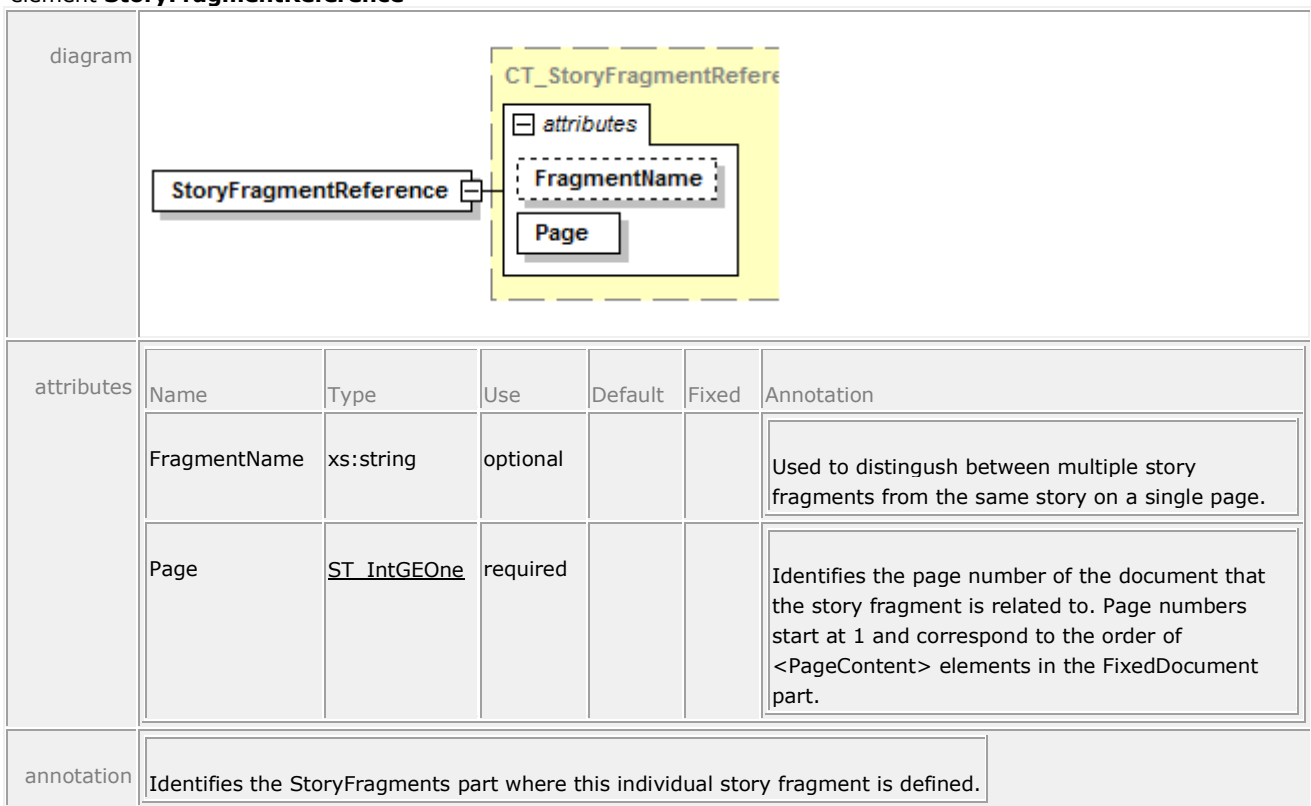
3 element **StoryFragments**



1 For more information, see §16.1.2.1.

2 19.67 StoryFragmentReference

3 element **StoryFragmentReference**



4 For more information, see §16.1.1.6.

5 19.68 TableCellStructure

6 element **TableCellStructure**

<p>diagram</p>	<p>The diagram shows a class CT_TableCell (highlighted in yellow) containing an attributes container with RowSpan and ColumnSpan attributes. It also contains a collection of child elements: ParagraphStructure, ListStructure, TableStructure, and FigureStructure. A TableCellStructure class is shown pointing to the CT_TableCell class. A multiplicity of 0..∞ is indicated for the child elements.</p>																						
<p>attributes</p>	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Use</th> <th>Default</th> <th>Fixed</th> <th>Annotation</th> </tr> </thead> <tbody> <tr> <td>RowSpan</td> <td><u>ST_TableSpan</u></td> <td>optional</td> <td>1</td> <td></td> <td>Indicates the number of rows this cell spans, or merges into a single cell.</td> </tr> <tr> <td>ColumnSpan</td> <td><u>ST_TableSpan</u></td> <td>optional</td> <td>1</td> <td></td> <td>Indicates the number of columns this cell spans, or merges into a single cell.</td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	RowSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of rows this cell spans, or merges into a single cell.	ColumnSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of columns this cell spans, or merges into a single cell.				
Name	Type	Use	Default	Fixed	Annotation																		
RowSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of rows this cell spans, or merges into a single cell.																		
ColumnSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of columns this cell spans, or merges into a single cell.																		
<p>annotation</p>	<p>Contains the elements that occupy a single cell of a table.</p>																						

1 For more information, see §16.1.2.9.

2 19.69 TableRowGroupStructure

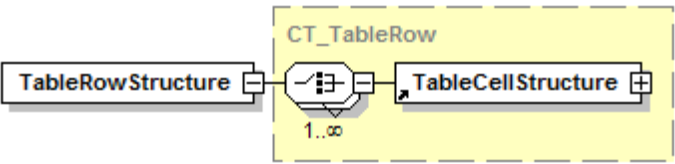
3 element **TableRowGroupStructure**

<p>diagram</p>	<p>The diagram shows a class CT_TableRowGroup (highlighted in yellow) containing a collection of TableRowStructure elements. A multiplicity of 1..∞ is indicated for the child elements.</p>				
<p>annotation</p>	<p>Contains the set of table rows that make up a table.</p>				

4 For more information, see §16.1.2.7.

5 19.70 TableRowStructure

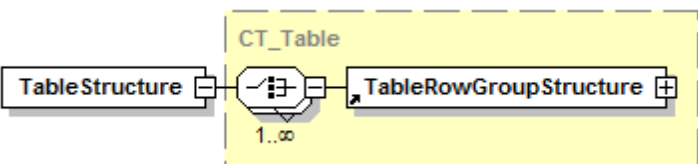
6 element **TableRowStructure**

<p>diagram</p>	 <p>The diagram shows a central container labeled 'CT_TableRow' with a dashed border. Inside this container, there is a 'TableRowStructure' box on the left and a 'TableCellStructure' box on the right. They are connected by a line with a central connector. Below the connector is the cardinality '1..∞'.</p>
<p>annotation</p>	<p>Contains the set of table cells that make up a row of a table.</p>

1 For more information, see §16.1.2.8.

2 19.71 TableStructure

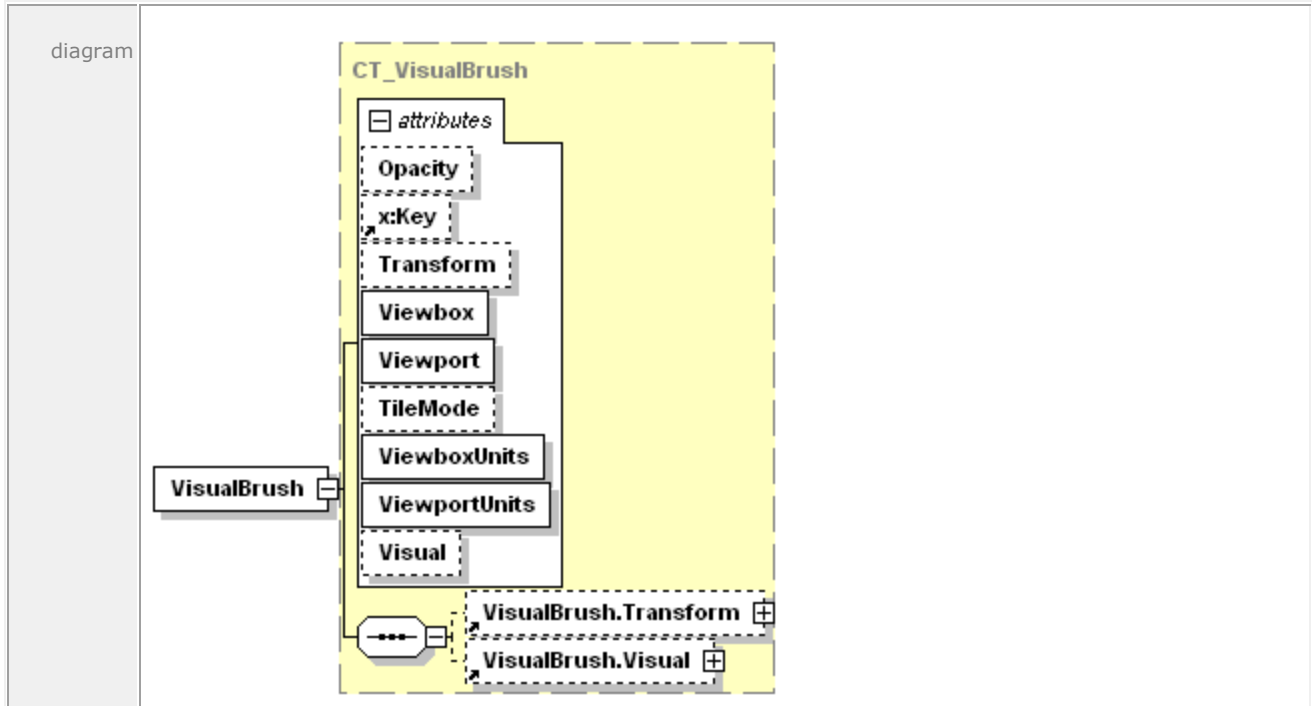
3 element **TableStructure**

<p>diagram</p>	 <p>The diagram shows a central container labeled 'CT_Table' with a dashed border. Inside this container, there is a 'TableStructure' box on the left and a 'TableRowGroupStructure' box on the right. They are connected by a line with a central connector. Below the connector is the cardinality '1..∞'.</p>
<p>annotation</p>	<p>Contains a complete definition of a table in the XPS Document.</p>

4 For more information, see §16.1.2.6.

5 19.72 VisualBrush

6 element **VisualBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	ST_ZeroOne		1.0		Defines the uniform transparency of the brush fill. Values range from 0 (fully transparent) to 1 (fully opaque), inclusive. Values outside of this range are invalid.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined in a resource dictionary. x:Key MUST NOT be specified outside of a resource dictionary [M6.4].
	Transform	ST_RscRefMatrix				Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using that local effective render transform.
	Viewbox	ST_ViewBox	required			Specifies the position and dimensions of the brush's source content. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The viewbox defines the default coordinate system for the element specified in the <code><VisualBrush.Visual></code> property element. The corners of the viewbox are mapped to the

					corners of the viewport, thereby providing the default clipping and transform for the brush's source content.
Viewport	ST_ViewBox	required			Specifies the region in the containing coordinate space of the prime brush tile that is (possibly repeatedly) applied to fill the region to which the brush is applied. Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative. The alignment of the brush pattern is controlled by adjusting the x and y values.
TileMode	ST_TileMode		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	ST_ViewUnits	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
Visual	ST_RscRef				Specifies resource reference to a <Path>, <Glyphs>, or <Canvas> element defined in a resource dictionary and used to draw the brush's source content.
annotation	Fills a region with a drawing. The drawing can be specified as either a child of the <VisualBrush> element, or as a resource reference. Drawing content is expressed using <Canvas>, <Path>, and <Glyphs> elements.				

1 For more information, see §13.3.

2 19.73 VisualBrush.Transform

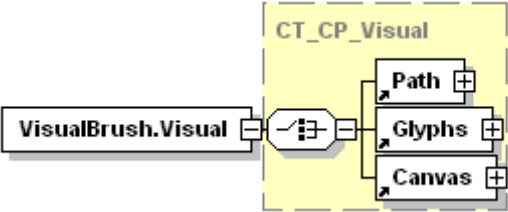
3 element **VisualBrush.Transform**

diagram	
annotation	Describes the matrix transformation applied to the coordinate space of the brush. The Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush. The viewport for the brush is transformed using the local effective render transform.

1 For more information, see §14.4.

2 **19.74 VisualBrush.Visual**

3 element **VisualBrush.Visual**

<p>diagram</p>	 <p>The diagram shows a class VisualBrush.Visual on the left. A solid line with an open arrowhead points from VisualBrush.Visual to a dashed-line box labeled CT_CP_Visual. Inside this dashed box, there are three stacked boxes: Path, Glyphs, and Canvas. Each of these three boxes has a small square icon with a plus sign in its top-right corner, indicating they are elements of the CT_CP_Visual container.</p>
<p>annotation</p>	<p>Specifies a <Path> element, <Glyphs> element, or <Canvas> element used to draw the brush's source contents.</p>

4

5

6

1 A. Signature Definitions W3C Schema

2 The schema shown below is also provided in electronic form as a file named
3 SignatureDefinitions.xsd, which is contained in an accompanying zip archive named "XPS WC3
4 Schemas.zip". If discrepancies exist between the representation as published below and the
5 corresponding electronic version, the published version below is the definitive version

```
6
7 <?xml version="1.0" encoding="utf-8"?>
8 <xs:schema targetNamespace="http://schemas.microsoft.com/xps/2005/06/signature-
9 definitions" xmlns="http://schemas.microsoft.com/xps/2005/06/signature-definitions"
10 xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
11 blockDefault="#all">
12
13   <xs:import namespace="http://www.w3.org/XML/1998/namespace" />
14
15   <xs:complexType name="SignatureDefinitionsType">
16     <xs:sequence>
17       <xs:element name="SignatureDefinition" type="SignatureDefinitionType"
18 minOccurs="1" maxOccurs="unbounded">
19         </xs:element>
20     </xs:sequence>
21   </xs:complexType>
22
23   <xs:complexType name="SpotLocationType">
24     <xs:attribute name="PageURI" type="xs:anyURI" use="required">
25       </xs:attribute>
26     <xs:attribute name="StartX" type="xs:double" use="required">
27       </xs:attribute>
28     <xs:attribute name="StartY" type="xs:double" use="required">
29       </xs:attribute>
30   </xs:complexType>
31
32   <xs:complexType name="SignatureDefinitionType">
33     <xs:sequence>
34       <xs:element name="SpotLocation" type="SpotLocationType" minOccurs="0">
35         </xs:element>
36       <xs:element name="Intent" type="xs:string" minOccurs="0">
37         </xs:element>
38       <xs:element name="SignBy" type="xs:dateTime" minOccurs="0">
39         </xs:element>
40       <xs:element name="SigningLocation" type="xs:string" minOccurs="0">
41         </xs:element>
42     </xs:sequence>
43     <xs:attribute name="SpotID" type="xs:ID" use="required">
44       </xs:attribute>
45     <xs:attribute name="SignerName" type="xs:string">
46       </xs:attribute>
47     <xs:attribute ref="xml:lang">
48       </xs:attribute>
49   </xs:complexType>
50
```

```
1     <xs:element name="SignatureDefinitions" type="SignatureDefinitionsType">
2         </xs:element>
3 </xs:schema>
4
```

1 B. XPS Document W3C Schema

2 The schema shown below is also provided in electronic form as a file named S0schema.xsd,
3 which is contained in an accompanying zip archive named "XPS WC3 Schemas.zip". If
4 discrepancies exist between the representation as published below and the corresponding
5 electronic version, the published version below is the definitive version

```
6
7 <?xml version="1.0" encoding="utf-8"?>
8 <xs:schema targetNamespace="http://schemas.microsoft.com/xps/2005/06"
9 xmlns="http://schemas.microsoft.com/xps/2005/06"
10 xmlns:mstns="http://tempuri.org/XMLSchema.xsd"
11 xmlns:xs="http://www.w3.org/2001/XMLSchema"
12 xmlns:x="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-key"
13 elementFormDefault="qualified" blockDefault="#all">
14
15     <xs:import namespace="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-
16 key" />
17     <xs:import namespace="http://www.w3.org/XML/1998/namespace" />
18
19     <!-- Names used for types and groups:
20
21         ST_*         simpleType
22         CT_*         complexType
23         G_*          group
24         AG_*         attributeGroup
25
26     -->
27
28     <!-- Individual real number patterns
29     All patterns using numbers now use <whitespace value="collapse">.
30     As a result, any whitespace in the pattern can be expressed as:
31     mandatory whitespace, one or more: " "
32     optional whitespace, zero or more: " ?"
33
34     For better readability, each pattern using numbers is also described in a comment
35     using
36     one of the following pattern designators.
37
38     The actual patterns are generated by replacement by the schema publication process.
39     -->
40     <!--DEFINE [pint]         "([1-9][0-9]*)" -->
41     <!--DEFINE [uint]        "([0-9]+)" -->
42     <!--DEFINE [dec]         "(\-?(((0-9)+(\.[0-9]+)?)|(\.[0-9]+)))" -->
43     <!--DEFINE [rn]         "((\-|\+)?((0-9)+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
44 9]+)?)" -->
45     <!--DEFINE [prn]        "(\+?(((0-9)+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
46 9]+)?)" -->
47     <!--DEFINE [scs]       "( ?, ?)" -->
48
49
50
```

```

1      <!-- Complex Types -->
2      <xs:complexType name="CT_MatrixTransform">
3          <xs:attributeGroup ref="AG_MatrixTransform" />
4      </xs:complexType>
5
6      <xs:complexType name="CT_SolidColorBrush">
7          <xs:attributeGroup ref="AG_Brush" />
8          <xs:attributeGroup ref="AG_SolidColorBrush" />
9      </xs:complexType>
10
11     <xs:complexType name="CT_ImageBrush">
12         <xs:sequence>
13             <xs:element ref="ImageBrush.Transform" minOccurs="0" />
14         </xs:sequence>
15         <xs:attributeGroup ref="AG_Brush" />
16         <xs:attributeGroup ref="AG_TileBrush" />
17         <xs:attributeGroup ref="AG_ImageBrush" />
18     </xs:complexType>
19
20     <xs:complexType name="CT_VisualBrush">
21         <xs:sequence>
22             <xs:element ref="VisualBrush.Transform" minOccurs="0" />
23             <xs:element ref="VisualBrush.Visual" minOccurs="0" />
24         </xs:sequence>
25         <xs:attributeGroup ref="AG_Brush" />
26         <xs:attributeGroup ref="AG_TileBrush" />
27         <xs:attributeGroup ref="AG_VisualBrush" />
28     </xs:complexType>
29
30     <xs:complexType name="CT_LinearGradientBrush">
31         <xs:sequence>
32             <xs:element ref="LinearGradientBrush.Transform" minOccurs="0" />
33             <xs:element ref="LinearGradientBrush.GradientStops" />
34         </xs:sequence>
35         <xs:attributeGroup ref="AG_Brush" />
36         <xs:attributeGroup ref="AG_GradientBrush" />
37         <xs:attributeGroup ref="AG_LinearGradientBrush" />
38     </xs:complexType>
39
40     <xs:complexType name="CT_RadialGradientBrush">
41         <xs:sequence>
42             <xs:element ref="RadialGradientBrush.Transform" minOccurs="0" />
43             <xs:element ref="RadialGradientBrush.GradientStops" />
44         </xs:sequence>
45         <xs:attributeGroup ref="AG_Brush" />
46         <xs:attributeGroup ref="AG_GradientBrush" />
47         <xs:attributeGroup ref="AG_RadialGradientBrush" />
48     </xs:complexType>
49
50     <xs:complexType name="CT_GradientStop">
51         <xs:attributeGroup ref="AG_GradientStop" />
52     </xs:complexType>
53
54     <xs:complexType name="CT_PathGeometry">
55         <xs:sequence>

```

```

1      <xs:element ref="PathGeometry.Transform" minOccurs="0" />
2      <xs:element ref="PathFigure" minOccurs="0" maxOccurs="unbounded" />
3    </xs:sequence>
4    <xs:attributeGroup ref="AG_PathGeometry" />
5  </xs:complexType>
6
7  <xs:complexType name="CT_Glyphs">
8    <xs:sequence>
9      <xs:element ref="Glyphs.RenderTransform" minOccurs="0" />
10     <xs:element ref="Glyphs.Clip" minOccurs="0" />
11     <xs:element ref="Glyphs.OpacityMask" minOccurs="0" />
12     <xs:element ref="Glyphs.Fill" minOccurs="0" />
13   </xs:sequence>
14   <xs:attributeGroup ref="AG_Glyphs" />
15 </xs:complexType>
16
17 <xs:complexType name="CT_Path">
18   <xs:sequence>
19     <xs:element ref="Path.RenderTransform" minOccurs="0" />
20     <xs:element ref="Path.Clip" minOccurs="0" />
21     <xs:element ref="Path.OpacityMask" minOccurs="0" />
22     <xs:element ref="Path.Fill" minOccurs="0" />
23     <xs:element ref="Path.Stroke" minOccurs="0" />
24     <xs:element ref="Path.Data" minOccurs="0" />
25   </xs:sequence>
26   <xs:attributeGroup ref="AG_Path" />
27   <xs:attributeGroup ref="AG_AutomationProvider" />
28   <xs:attributeGroup ref="AG_SnapsToDevicePixels" />
29 </xs:complexType>
30
31 <xs:complexType name="CT_PathFigure">
32   <xs:sequence>
33     <xs:choice maxOccurs="unbounded">
34       <xs:element ref="PolyLineSegment" />
35       <xs:element ref="PolyBezierSegment" />
36       <xs:element ref="ArcSegment" />
37       <xs:element ref="PolyQuadraticBezierSegment" />
38     </xs:choice>
39   </xs:sequence>
40   <xs:attributeGroup ref="AG_PathFigure" />
41 </xs:complexType>
42
43 <xs:complexType name="CT_ArcSegment">
44   <xs:attributeGroup ref="AG_ArcSegment" />
45 </xs:complexType>
46
47 <xs:complexType name="CT_PolyQuadraticBezierSegment">
48   <xs:attributeGroup ref="AG_PolyQuadraticBezierSegment" />
49 </xs:complexType>
50
51 <xs:complexType name="CT_PolyLineSegment">
52   <xs:attributeGroup ref="AG_PolyLineSegment" />
53 </xs:complexType>
54
55 <xs:complexType name="CT_PolyBezierSegment">

```

```

1      <xs:attributeGroup ref="AG_PolyBezierSegment" />
2  </xs:complexType>
3
4  <xs:complexType name="CT_Canvas">
5    <xs:sequence>
6      <xs:element ref="Canvas.Resources" minOccurs="0" />
7      <xs:element ref="Canvas.RenderTransform" minOccurs="0" />
8      <xs:element ref="Canvas.Clip" minOccurs="0" />
9      <xs:element ref="Canvas.OpacityMask" minOccurs="0" />
10     <xs:choice minOccurs="0" maxOccurs="unbounded">
11       <xs:element ref="Path" />
12       <xs:element ref="Glyphs" />
13       <xs:element ref="Canvas" />
14     </xs:choice>
15   </xs:sequence>
16   <xs:attributeGroup ref="AG_Canvas" />
17   <xs:attributeGroup ref="AG_AutomationProvider" />
18 </xs:complexType>
19
20 <xs:complexType name="CT_ResourceDictionary">
21   <xs:choice minOccurs="0" maxOccurs="unbounded">
22     <xs:element ref="ImageBrush" />
23     <xs:element ref="LinearGradientBrush" />
24     <xs:element ref="RadialGradientBrush" />
25     <xs:element ref="VisualBrush" />
26     <xs:element ref="SolidColorBrush" />
27     <xs:element ref="MatrixTransform" />
28     <xs:element ref="PathGeometry" />
29     <xs:element ref="Path" />
30     <xs:element ref="Glyphs" />
31     <xs:element ref="Canvas" />
32   </xs:choice>
33   <xs:attributeGroup ref="AG_ResourceDictionary" />
34 </xs:complexType>
35
36 <xs:complexType name="CT_FixedPage">
37   <xs:sequence>
38     <xs:element ref="FixedPage.Resources" minOccurs="0" />
39     <xs:choice minOccurs="0" maxOccurs="unbounded">
40       <xs:element ref="Path" />
41       <xs:element ref="Glyphs" />
42       <xs:element ref="Canvas" />
43     </xs:choice>
44   </xs:sequence>
45   <xs:attributeGroup ref="AG_FixedPage" />
46 </xs:complexType>
47
48 <xs:complexType name="CT_FixedDocument">
49   <xs:sequence>
50     <xs:element ref="PageContent" maxOccurs="unbounded" />
51   </xs:sequence>
52 </xs:complexType>
53
54 <xs:complexType name="CT_PageContent">
55   <xs:sequence>

```



```
1         <xs:element ref="PageContent.LinkTargets" minOccurs="0" />
2     </xs:sequence>
3     <xs:attributeGroup ref="AG_PageContent" />
4 </xs:complexType>
5
6 <xs:complexType name="CT_FixedDocumentSequence">
7     <xs:sequence>
8         <xs:element ref="DocumentReference" maxOccurs="unbounded" />
9     </xs:sequence>
10 </xs:complexType>
11
12 <xs:complexType name="CT_DocumentReference">
13     <xs:attributeGroup ref="AG_DocumentReference" />
14 </xs:complexType>
15
16 <xs:complexType name="CT_LinkTarget">
17     <xs:attributeGroup ref="AG_LinkTarget" />
18 </xs:complexType>
19
20 <xs:complexType name="CT_CP_LinkTargets">
21     <xs:sequence>
22         <xs:element ref="LinkTarget" maxOccurs="unbounded" />
23     </xs:sequence>
24 </xs:complexType>
25
26 <xs:complexType name="CT_CP_Transform">
27     <xs:sequence>
28         <xs:element ref="MatrixTransform" />
29     </xs:sequence>
30 </xs:complexType>
31
32 <xs:complexType name="CT_CP_Visual">
33     <xs:choice>
34         <xs:element ref="Path" />
35         <xs:element ref="Glyphs" />
36         <xs:element ref="Canvas" />
37     </xs:choice>
38 </xs:complexType>
39
40 <xs:complexType name="CT_CP_GradientStops">
41     <xs:sequence>
42         <xs:element ref="GradientStop" minOccurs="2" maxOccurs="unbounded" />
43     </xs:sequence>
44 </xs:complexType>
45
46 <xs:complexType name="CT_CP_Geometry">
47     <xs:sequence>
48         <xs:element ref="PathGeometry" />
49     </xs:sequence>
50 </xs:complexType>
51
52 <xs:complexType name="CT_CP_Brush">
53     <xs:choice>
54         <xs:element ref="ImageBrush" />
55         <xs:element ref="LinearGradientBrush" />
```

```

1         <xs:element ref="RadialGradientBrush" />
2         <xs:element ref="SolidColorBrush" />
3         <xs:element ref="VisualBrush" />
4     </xs:choice>
5 </xs:complexType>
6
7 <xs:complexType name="CT_CP_Resources">
8     <xs:sequence minOccurs="0">
9         <xs:element ref="ResourceDictionary" />
10    </xs:sequence>
11 </xs:complexType>
12
13 <!-- Root elements -->
14 <xs:element name="MatrixTransform" type="CT_MatrixTransform">
15     </xs:element>
16
17 <xs:element name="SolidColorBrush" type="CT_SolidColorBrush">
18     </xs:element>
19
20 <xs:element name="ImageBrush" type="CT_ImageBrush">
21     </xs:element>
22
23 <xs:element name="VisualBrush" type="CT_VisualBrush">
24     </xs:element>
25
26 <xs:element name="LinearGradientBrush" type="CT_LinearGradientBrush">
27     </xs:element>
28
29 <xs:element name="RadialGradientBrush" type="CT_RadialGradientBrush">
30     </xs:element>
31
32 <xs:element name="Glyphs" type="CT_Glyphs">
33     </xs:element>
34
35 <xs:element name="Path" type="CT_Path">
36     </xs:element>
37
38 <xs:element name="Canvas" type="CT_Canvas">
39     </xs:element>
40
41 <xs:element name="GradientStop" type="CT_GradientStop">
42     </xs:element>
43
44 <xs:element name="ResourceDictionary" type="CT_ResourceDictionary">
45     </xs:element>
46
47 <xs:element name="PathGeometry" type="CT_PathGeometry">
48     </xs:element>
49
50 <xs:element name="PathFigure" type="CT_PathFigure">
51     </xs:element>
52
53 <xs:element name="PolyLineSegment" type="CT_PolyLineSegment">
54     </xs:element>
55

```

```
1      <xs:element name="ArcSegment" type="CT_ArcSegment">
2          </xs:element>
3
4      <xs:element name="PolyBezierSegment" type="CT_PolyBezierSegment">
5          </xs:element>
6
7      <xs:element name="PolyQuadraticBezierSegment" type="CT_PolyQuadraticBezierSegment">
8          </xs:element>
9
10     <xs:element name="FixedPage" type="CT_FixedPage">
11         </xs:element>
12
13     <xs:element name="FixedDocument" type="CT_FixedDocument">
14         </xs:element>
15
16     <xs:element name="PageContent" type="CT_PageContent">
17         </xs:element>
18
19     <xs:element name="FixedDocumentSequence" type="CT_FixedDocumentSequence">
20         </xs:element>
21
22     <xs:element name="DocumentReference" type="CT_DocumentReference">
23         </xs:element>
24
25     <xs:element name="LinkTarget" type="CT_LinkTarget">
26         </xs:element>
27
28     <xs:element name="PageContent.LinkTargets" type="CT_CP_LinkTargets">
29         </xs:element>
30
31     <xs:element name="ImageBrush.Transform" type="CT_CP_Transform">
32         </xs:element>
33
34     <xs:element name="VisualBrush.Transform" type="CT_CP_Transform">
35         </xs:element>
36
37     <xs:element name="LinearGradientBrush.Transform" type="CT_CP_Transform">
38         </xs:element>
39
40     <xs:element name="RadialGradientBrush.Transform" type="CT_CP_Transform">
41         </xs:element>
42
43     <xs:element name="PathGeometry.Transform" type="CT_CP_Transform">
44         </xs:element>
45
46     <xs:element name="Glyphs.RenderTransform" type="CT_CP_Transform">
47         </xs:element>
48
49     <xs:element name="Path.RenderTransform" type="CT_CP_Transform">
50         </xs:element>
51
52     <xs:element name="Canvas.RenderTransform" type="CT_CP_Transform">
53         </xs:element>
54
55     <xs:element name="VisualBrush.Visual" type="CT_CP_Visual">
```

```
1         </xs:element>
2
3     <xs:element name="LinearGradientBrush.GradientStops" type="CT_CP_GradientStops">
4         </xs:element>
5
6     <xs:element name="RadialGradientBrush.GradientStops" type="CT_CP_GradientStops">
7         </xs:element>
8
9     <xs:element name="Glyphs.Clip" type="CT_CP_Geometry">
10        </xs:element>
11
12    <xs:element name="Path.Clip" type="CT_CP_Geometry">
13        </xs:element>
14
15    <xs:element name="Canvas.Clip" type="CT_CP_Geometry">
16        </xs:element>
17
18    <xs:element name="Glyphs.OpacityMask" type="CT_CP_Brush">
19        </xs:element>
20
21    <xs:element name="Path.OpacityMask" type="CT_CP_Brush">
22        </xs:element>
23
24    <xs:element name="Canvas.OpacityMask" type="CT_CP_Brush">
25        </xs:element>
26
27    <xs:element name="Glyphs.Fill" type="CT_CP_Brush">
28        </xs:element>
29
30    <xs:element name="Path.Fill" type="CT_CP_Brush">
31        </xs:element>
32
33    <xs:element name="Path.Data" type="CT_CP_Geometry">
34        </xs:element>
35
36    <xs:element name="Path.Stroke" type="CT_CP_Brush">
37        </xs:element>
38
39    <xs:element name="Canvas.Resources" type="CT_CP_Resources">
40        </xs:element>
41
42    <xs:element name="FixedPage.Resources" type="CT_CP_Resources">
43        </xs:element>
44
45    <xs:attributeGroup name="AG_GradientStop">
46        <xs:attribute name="Color" type="ST_Color" use="required">
47            </xs:attribute>
48        <xs:attribute name="Offset" type="ST_Double" use="required">
49            </xs:attribute>
50    </xs:attributeGroup>
51
52    <xs:attributeGroup name="AG_Brush">
53        <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
54            </xs:attribute>
55        <xs:attribute ref="x:Key" />
```

```
1     </xs:attributeGroup>
2
3     <xs:attributeGroup name="AG_TileBrush">
4         <xs:attribute name="Transform" type="ST_RscRefMatrix">
5             </xs:attribute>
6         <xs:attribute name="Viewbox" type="ST_ViewBox" use="required">
7             </xs:attribute>
8         <xs:attribute name="Viewport" type="ST_ViewBox" use="required">
9             </xs:attribute>
10        <xs:attribute name="TileMode" type="ST_TileMode" default="None">
11            </xs:attribute>
12        <xs:attribute name="ViewboxUnits" type="ST_ViewUnits" use="required"
13fixed="Absolute">
14            </xs:attribute>
15        <xs:attribute name="ViewportUnits" type="ST_ViewUnits" use="required"
16fixed="Absolute">
17            </xs:attribute>
18    </xs:attributeGroup>
19
20    <xs:attributeGroup name="AG_VisualBrush">
21        <xs:attribute name="Visual" type="ST_RscRef">
22            </xs:attribute>
23    </xs:attributeGroup>
24
25    <xs:attributeGroup name="AG_GradientBrush">
26        <xs:attribute name="ColorInterpolationMode" type="ST_ClrIntMode"
27default="SRgbLinearInterpolation">
28            </xs:attribute>
29        <xs:attribute name="SpreadMethod" type="ST_SpreadMethod" default="Pad">
30            </xs:attribute>
31        <xs:attribute name="MappingMode" type="ST_MappingMode" use="required"
32fixed="Absolute">
33            </xs:attribute>
34    </xs:attributeGroup>
35
36    <xs:attributeGroup name="AG_SolidColorBrush">
37        <xs:attribute name="Color" type="ST_Color" use="required">
38            </xs:attribute>
39    </xs:attributeGroup>
40
41    <xs:attributeGroup name="AG_ImageBrush">
42        <xs:attribute name="ImageSource" type="ST_UniCtxBmp" use="required">
43            </xs:attribute>
44    </xs:attributeGroup>
45
46    <xs:attributeGroup name="AG_LinearGradientBrush">
47        <xs:attribute name="Transform" type="ST_RscRefMatrix">
48            </xs:attribute>
49        <xs:attribute name="StartPoint" type="ST_Point" use="required">
50            </xs:attribute>
51        <xs:attribute name="EndPoint" type="ST_Point" use="required">
52            </xs:attribute>
53    </xs:attributeGroup>
54
55    <xs:attributeGroup name="AG_RadialGradientBrush">
```

```

1      <xs:attribute name="Transform" type="ST_RscRefMatrix">
2          </xs:attribute>
3      <xs:attribute name="Center" type="ST_Point" use="required">
4          </xs:attribute>
5      <xs:attribute name="GradientOrigin" type="ST_Point" use="required">
6          </xs:attribute>
7      <xs:attribute name="RadiusX" type="ST_GEZero" use="required">
8          </xs:attribute>
9      <xs:attribute name="RadiusY" type="ST_GEZero" use="required">
10         </xs:attribute>
11     </xs:attributeGroup>
12
13     <xs:attributeGroup name="AG_PathGeometry">
14         <xs:attribute name="Figures" type="ST_AbbrGeom">
15             </xs:attribute>
16         <xs:attribute name="FillRule" type="ST_FillRule" default="EvenOdd">
17             </xs:attribute>
18         <xs:attribute name="Transform" type="ST_RscRefMatrix">
19             </xs:attribute>
20         <xs:attribute ref="x:Key" />
21     </xs:attributeGroup>
22
23     <xs:attributeGroup name="AG_ResourceDictionary">
24         <xs:attribute name="Source" type="xs:anyURI">
25             </xs:attribute>
26     </xs:attributeGroup>
27
28     <xs:attributeGroup name="AG_PolyLineSegment">
29         <xs:attribute name="Points" type="ST_Points" use="required">
30             </xs:attribute>
31         <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
32             </xs:attribute>
33     </xs:attributeGroup>
34
35     <xs:attributeGroup name="AG_ArcSegment">
36         <xs:attribute name="Point" type="ST_Point" use="required">
37             </xs:attribute>
38         <xs:attribute name="Size" type="ST_PointGE0" use="required">
39             </xs:attribute>
40         <xs:attribute name="RotationAngle" type="ST_Double" use="required">
41             </xs:attribute>
42         <xs:attribute name="IsLargeArc" type="ST_Boolean" use="required">
43             </xs:attribute>
44         <xs:attribute name="SweepDirection" type="ST_SweepDirection" use="required">
45             </xs:attribute>
46         <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
47             </xs:attribute>
48     </xs:attributeGroup>
49
50     <xs:attributeGroup name="AG_PolyBezierSegment">
51         <xs:attribute name="Points" type="ST_Points" use="required">
52             </xs:attribute>
53         <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
54             </xs:attribute>
55     </xs:attributeGroup>

```

```

1
2     <xs:attributeGroup name="AG_PolyQuadraticBezierSegment">
3         <xs:attribute name="Points" type="ST_Points" use="required">
4             </xs:attribute>
5         <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
6             </xs:attribute>
7     </xs:attributeGroup>
8
9     <xs:attributeGroup name="AG_Glyphs">
10        <xs:attribute name="BidiLevel" default="0">
11            <xs:simpleType>
12                <xs:restriction base="xs:integer">
13                    <xs:minInclusive value="0" />
14                    <xs:maxInclusive value="61" />
15                </xs:restriction>
16            </xs:simpleType>
17        </xs:attribute>
18        <xs:attribute name="CaretStops" type="ST_CaretStops">
19            </xs:attribute>
20        <xs:attribute name="DeviceFontName" type="ST_UnicodeString">
21            </xs:attribute>
22        <xs:attribute name="Fill" type="ST_RscRefColor">
23            </xs:attribute>
24        <xs:attribute name="FontRenderingEmSize" type="ST_GEZero" use="required">
25            </xs:attribute>
26        <xs:attribute name="FontUri" type="xs:anyURI" use="required">
27            </xs:attribute>
28        <xs:attribute name="OriginX" type="ST_Double" use="required">
29            </xs:attribute>
30        <xs:attribute name="OriginY" type="ST_Double" use="required">
31            </xs:attribute>
32        <xs:attribute name="IsSideways" type="ST_Boolean" default="false">
33            </xs:attribute>
34        <xs:attribute name="Indices" type="ST_Indices">
35            </xs:attribute>
36        <xs:attribute name="UnicodeString" type="ST_UnicodeString">
37            </xs:attribute>
38        <xs:attribute name="StyleSimulations" type="ST_StyleSimulations"
39 default="None">
40            </xs:attribute>
41        <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">
42            </xs:attribute>
43        <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">
44            </xs:attribute>
45        <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
46            </xs:attribute>
47        <xs:attribute name="OpacityMask" type="ST_RscRef">
48            </xs:attribute>
49        <xs:attribute name="Name" type="ST_Name">
50            </xs:attribute>
51        <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
52            </xs:attribute>
53        <xs:attribute ref="xml:lang">
54            </xs:attribute>
55        <xs:attribute ref="x:Key" />

```

```

1      </xs:attributeGroup>
2
3      <xs:attributeGroup name="AG_Path">
4          <xs:attribute name="Data" type="ST_RscRefAbbrGeomF">
5              </xs:attribute>
6          <xs:attribute name="Fill" type="ST_RscRefColor">
7              </xs:attribute>
8          <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">
9              </xs:attribute>
10         <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">
11             </xs:attribute>
12         <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
13             </xs:attribute>
14         <xs:attribute name="OpacityMask" type="ST_RscRef">
15             </xs:attribute>
16         <xs:attribute name="Stroke" type="ST_RscRefColor">
17             </xs:attribute>
18         <xs:attribute name="StrokeDashArray" type="ST_EvenArrayPos">
19             </xs:attribute>
20         <xs:attribute name="StrokeDashCap" type="ST_DashCap" default="Flat">
21             </xs:attribute>
22         <xs:attribute name="StrokeDashOffset" type="ST_Double" default="0.0">
23             </xs:attribute>
24         <xs:attribute name="StrokeEndLineCap" type="ST_LineCap" default="Flat">
25             </xs:attribute>
26         <xs:attribute name="StrokeStartLineCap" type="ST_LineCap" default="Flat">
27             </xs:attribute>
28         <xs:attribute name="StrokeLineJoin" type="ST_LineJoin" default="Miter">
29             </xs:attribute>
30         <xs:attribute name="StrokeMiterLimit" type="ST_GEOne" default="10.0">
31             </xs:attribute>
32         <xs:attribute name="StrokeThickness" type="ST_GEZero" default="1.0">
33             </xs:attribute>
34         <xs:attribute name="Name" type="ST_Name">
35             </xs:attribute>
36         <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
37             </xs:attribute>
38         <xs:attribute ref="xml:lang">
39             </xs:attribute>
40         <xs:attribute ref="x:Key" />
41     </xs:attributeGroup>
42
43     <xs:attributeGroup name="AG_PathFigure">
44         <xs:attribute name="IsClosed" type="ST_Boolean" default="false">
45             </xs:attribute>
46         <xs:attribute name="StartPoint" type="ST_Point" use="required">
47             </xs:attribute>
48         <xs:attribute name="IsFilled" type="ST_Boolean" default="true">
49             </xs:attribute>
50     </xs:attributeGroup>
51
52     <xs:attributeGroup name="AG_Canvas">
53         <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">
54             </xs:attribute>
55         <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">

```



```
1         </xs:attribute>
2     <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
3         </xs:attribute>
4     <xs:attribute name="OpacityMask" type="ST_RscRef">
5         </xs:attribute>
6     <xs:attribute name="Name" type="ST_Name">
7         </xs:attribute>
8     <xs:attribute name="RenderOptions.EdgeMode" type="ST_EdgeMode">
9         </xs:attribute>
10    <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
11        </xs:attribute>
12    <xs:attribute ref="xml:lang">
13        </xs:attribute>
14    <xs:attribute ref="x:Key" />
15 </xs:attributeGroup>
16
17 <xs:attributeGroup name="AG_PageContent">
18     <xs:attribute name="Source" type="xs:anyURI" use="required">
19         </xs:attribute>
20     <xs:attribute name="Width" type="ST_GEOne">
21         </xs:attribute>
22     <xs:attribute name="Height" type="ST_GEOne">
23         </xs:attribute>
24 </xs:attributeGroup>
25
26 <xs:attributeGroup name="AG_LinkTarget">
27     <xs:attribute name="Name" type="ST_NUName" use="required">
28         </xs:attribute>
29 </xs:attributeGroup>
30
31 <xs:attributeGroup name="AG_DocumentReference">
32     <xs:attribute name="Source" type="xs:anyURI" use="required">
33         </xs:attribute>
34 </xs:attributeGroup>
35
36 <xs:attributeGroup name="AG_MatrixTransform">
37     <xs:attribute name="Matrix" type="ST_Matrix" use="required">
38         </xs:attribute>
39     <xs:attribute ref="x:Key" />
40 </xs:attributeGroup>
41
42 <xs:attributeGroup name="AG_FixedPage">
43     <xs:attribute name="Width" type="ST_GEOne" use="required">
44         </xs:attribute>
45     <xs:attribute name="Height" type="ST_GEOne" use="required">
46         </xs:attribute>
47     <xs:attribute name="ContentBox" type="ST_ContentBox">
48         </xs:attribute>
49     <xs:attribute name="BleedBox" type="ST_BleedBox">
50         </xs:attribute>
51     <xs:attribute ref="xml:lang" use="required">
52         </xs:attribute>
53     <xs:attribute name="Name" type="ST_Name">
54         </xs:attribute>
55 </xs:attributeGroup>
```

```

1
2     <xs:attributeGroup name="AG_AutomationProvider">
3         <xs:attribute name="AutomationProperties.Name" type="xs:string">
4             </xs:attribute>
5         <xs:attribute name="AutomationProperties.HelpText" type="xs:string">
6             </xs:attribute>
7     </xs:attributeGroup>
8
9     <xs:attributeGroup name="AG_SnapsToDevicePixels">
10        <xs:attribute name="SnapsToDevicePixels" type="ST_Boolean">
11            </xs:attribute>
12    </xs:attributeGroup>
13
14    <!-- Simple data types -->
15    <!-- A unique Name (ID with pattern restriction according to XPS spec) -->
16    <xs:simpleType name="ST_Name">
17        <xs:restriction base="xs:ID">
18            <xs:pattern
19 value="(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|_)(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|\p{Mn}
20 |\p{Mc}|\p{Nd}|_)*" />
21        </xs:restriction>
22    </xs:simpleType>
23
24    <!-- A non-unique Name (ID with pattern restriction according to XPS spec) -->
25    <xs:simpleType name="ST_NUName">
26        <xs:restriction base="xs:string">
27            <xs:pattern
28 value="(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|_)(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|\p{Mn}
29 |\p{Mc}|\p{Nd}|_)*" />
30        </xs:restriction>
31    </xs:simpleType>
32
33    <!-- Boolean with true and false only (no 0 or 1) -->
34    <xs:simpleType name="ST_Boolean">
35        <xs:restriction base="xs:boolean">
36            <xs:pattern value="true|false" />
37        </xs:restriction>
38    </xs:simpleType>
39
40    <!-- real number from 0.0 to 1.0 inclusive -->
41    <xs:simpleType name="ST_ZeroOne">
42        <xs:restriction base="ST_Double">
43            <xs:minInclusive value="0.0" />
44            <xs:maxInclusive value="1.0" />
45        </xs:restriction>
46    </xs:simpleType>
47
48    <!-- positive real number -->
49    <xs:simpleType name="ST_GEZero">
50        <xs:restriction base="ST_Double">
51            <xs:minInclusive value="0.0" />
52        </xs:restriction>
53    </xs:simpleType>
54
55    <!-- positive real number, equal or greater than one -->

```

```

1      <xs:simpleType name="ST_GEOne">
2          <xs:restriction base="ST_Double">
3              <xs:minInclusive value="1.0" />
4          </xs:restriction>
5      </xs:simpleType>
6
7      <!-- Double -->
8      <xs:simpleType name="ST_Double">
9          <xs:restriction base="xs:double">
10             <xs:whiteSpace value="collapse" />
11 <!--
12             <xs:pattern value="[rn]" />
13 -->
14             <xs:pattern value="((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
15 |\+)?[0-9]+)?)" />
16         </xs:restriction>
17     </xs:simpleType>
18
19     <!-- Point: 2 numbers, separated by , and arbitrary whitespace -->
20     <xs:simpleType name="ST_Point">
21         <xs:restriction base="xs:string">
22             <xs:whiteSpace value="collapse" />
23 <!--
24             <xs:pattern value="[rn][scs][rn]" />
25 -->
26             <xs:pattern value="((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
27 |\+)?[0-9]+)?)( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)"
28 />
29         </xs:restriction>
30     </xs:simpleType>
31
32     <!-- PointGE0: 2 non-negative numbers, separated by , and arbitrary whitespace -->
33     <xs:simpleType name="ST_PointGE0">
34         <xs:restriction base="xs:string">
35             <xs:whiteSpace value="collapse" />
36 <!--
37             <xs:pattern value="[prn][scs][prn]" />
38 -->
39             <xs:pattern value="(\+?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
40 9]+)?)( ?, ?)(\+?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />
41         </xs:restriction>
42     </xs:simpleType>
43
44     <!-- Points: List of ST_Point, separated by arbitrary whitespace -->
45     <xs:simpleType name="ST_Points">
46         <xs:restriction base="xs:string">
47             <xs:whiteSpace value="collapse" />
48 <!--
49             <xs:pattern value="[rn][scs][rn]( [rn][scs][rn])*" />
50 -->
51             <xs:pattern value="((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
52 |\+)?[0-9]+)?)( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)(
53 ((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ?, ?)((\-|\+)?((([0-
54 9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))*" />
55         </xs:restriction>

```

```

1      </xs:simpleType>
2
3      <!-- EvenArray: List with even number of entries of non-negative numbers. -->
4      <xs:simpleType name="ST_EvenArrayPos">
5          <xs:restriction base="xs:string">
6              <xs:whiteSpace value="collapse" />
7      <!--
8          <xs:pattern value="[prn] [prn]( [prn] [prn])*"/>
9      -->
10         <xs:pattern value="(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
11 9]+)?)(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)(\+?(([0-9]+(\.[0-
12 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
13 |\+)?[0-9]+)?))*" />
14     </xs:restriction>
15 </xs:simpleType>
16
17 <!-- Array: List of numbers. -->
18 <xs:simpleType name="ST_Array">
19     <xs:restriction base="xs:string">
20         <xs:whiteSpace value="collapse" />
21 <!--
22     <xs:pattern value="([rn] ?)*"/>
23 -->
24     <xs:pattern value="(((\-|\+)?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
25 |\+)?[0-9]+)?) ?)*" />
26     </xs:restriction>
27 </xs:simpleType>
28
29 <!-- ViewBox: 4 numbers, separated by , and arbitrary whitespace. Second number
30 pair must be non-negative -->
31 <xs:simpleType name="ST_ViewBox">
32     <xs:restriction base="xs:string">
33         <xs:whiteSpace value="collapse" />
34 <!--
35     <xs:pattern value="[rn][scs][rn][scs][prn][scs][prn]"/>
36 -->
37     <xs:pattern value="(((\-|\+)?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
38 |\+)?[0-9]+)?)( ?, ?)((\-|\+)?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)(
39 ?, ?)(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ?, ?)(\+?(([0-
40 9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />
41     </xs:restriction>
42 </xs:simpleType>
43
44 <!-- ContentBox: 4 non-negative numbers, separated by commas and arbitrary
45 whitespace -->
46 <xs:simpleType name="ST_ContentBox">
47     <xs:restriction base="xs:string">
48         <xs:whiteSpace value="collapse" />
49 <!--
50     <xs:pattern value="[prn][scs][prn][scs][prn][scs][prn]"/>
51 -->
52     <xs:pattern value="(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
53 9]+)?)( ?, ?)(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ?,
54 ?)(\+?(([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ?, ?)(\+?(([0-9]+(\.[0-
55 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />

```

```

1      </xs:restriction>
2      </xs:simpleType>
3
4      <!-- BleedBox: 4 numbers, separated by , and arbitrary whitespace. Second number
5      pair must be non-negative -->
6      <xs:simpleType name="ST_BleedBox">
7      <xs:restriction base="xs:string">
8      <xs:whiteSpace value="collapse" />
9      <!--
10     <xs:pattern value="[rn][scs][rn][scs][prn][scs][prn]" />
11     -->
12     <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
13     |\+)?[\0-9]+)?)( ? , ?)(\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)(
14     ? , ?)(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ? , ?)(\+?(([\0-
15     9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)" />
16     </xs:restriction>
17     </xs:simpleType>
18
19     <!-- Bare Matrix form: 6 numbers separated by , and arbitrary whitespace -->
20     <xs:simpleType name="ST_Matrix">
21     <xs:restriction base="xs:string">
22     <xs:whiteSpace value="collapse" />
23     <!--
24     <xs:pattern value="[rn][scs][rn][scs][rn][scs][rn][scs][rn][scs][rn]" />
25     -->
26     <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
27     |\+)?[\0-9]+)?)( ? , ?)(\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)(
28     ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ? , ?)(\-
29     |\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ? , ?)(\-|\+)?(([\0-
30     9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ? , ?)(\-|\+)?(([\0-9]+(\.[\0-
31     9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)" />
32     </xs:restriction>
33     </xs:simpleType>
34
35     <!-- Color: 6 or 8 hex digits -->
36     <xs:simpleType name="ST_Color">
37     <xs:restriction base="xs:string">
38     <!-- The pattern restriction does not check for sRGB gamut -->
39     <!-- The pattern restriction does not check for color profile URI validity
40     -->
41     <xs:whiteSpace value="collapse" />
42     <!--
43     <xs:pattern value="(#[\0-9a-fA-F]{2})?[\0-9a-fA-F]{6})|\
44     (sc# ?[dec][scs][dec][scs][dec]([scs][dec])?)|\
45     (ContextColor +[\S]+ ?[dec]([scs][dec]){3,8})" />
46     -->
47     <xs:pattern value="(#[\0-9a-fA-F]{2})?[\0-9a-fA-F]{6})|(sc# ?(\-?(([\0-
48     9]+(\.[\0-9]+)?)|(\.[\0-9]+))) ( ? , ?)(\-?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))) ( ? , ?)(\-
49     ?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))) (( ? , ?)(\-?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-
50     9]+))))?|(ContextColor +[\S]+ ?(\-?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))) (( ? , ?)(\-?(([\0-
51     9]+(\.[\0-9]+)?)|(\.[\0-9]+))))){3,8})" />
52     </xs:restriction>
53     </xs:simpleType>
54
55     <!-- Indices grammar for Glyphs.CaretStops -->

```

```

1      <xs:simpleType name="ST_CaretStops">
2          <xs:restriction base="xs:string">
3              <xs:whiteSpace value="collapse" />
4              <xs:pattern value="[0-9A-Fa-f]*" />
5          </xs:restriction>
6      </xs:simpleType>
7
8      <!-- Indices grammar for Glyphs.Indices -->
9      <xs:simpleType name="ST_Indices">
10         <xs:restriction base="xs:string">
11             <xs:whiteSpace value="collapse" />
12     <!--
13         <xs:pattern value="(\
14             ((\[pint](:[pint])?\))?[uint])?\
15             (,[prn]?([rn]?([rn])?)?)?\
16             )\
17             (;\
18             ((\[pint](:[pint])?\))?[uint])?\
19             (,[prn]?([rn]?([rn])?)?)?\
20             )*" />
21     -->
22         <xs:pattern value="(((\[0-9]*)(:[0-9]*)?)?([0-
23 9]+)?(,( \+?([0-9]+(\.[0-9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?),((\-|\+)?([0-
24 9]+(\.[0-9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?),((\-|\+)?([0-9]+(\.[0-
25 9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))?)?);(((\[0-9]*)(:[0-9]*)?)(:[0-
26 9]*)?)?([0-9]+)?(,( \+?([0-9]+(\.[0-9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-
27 9]+)?),((\-|\+)?([0-9]+(\.[0-9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?),((\-
28 |\+)?([0-9]+(\.[0-9]+)?|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))?)?" />
29     </xs:restriction>
30 </xs:simpleType>
31
32     <!-- UnicodeString grammar -->
33     <xs:simpleType name="ST_UnicodeString">
34         <xs:restriction base="xs:string">
35             <xs:pattern value="(([\^{}]|(\{\})).*)" />
36         </xs:restriction>
37     </xs:simpleType>
38
39     <!-- Abbreviated Geometry grammar for Path.Data , clip and Geometries -->
40     <xs:simpleType name="ST_AbbrGeomF">
41         <xs:restriction base="xs:string">
42             <xs:whiteSpace value="collapse" />
43     <!--
44         <xs:pattern value="(F ?(0|1))?\
45             ( ?(M|m)( ?[rn][scs][rn]))\
46             (\
47                 ( ?(M|m)( ?[rn][scs][rn]))|\
48                 ( ?(L|l)( ?[rn][scs][rn])( [rn][scs][rn])*)|\
49                 ( ?(H|h|V|v)( ?[rn])( [rn])*)|\
50                 ( ?(Q|q|S|s)( ?[rn][scs][rn] [rn][scs][rn])((
51 [rn][scs][rn]){2})*)|\
52                 ( ?(C|c)( ?[rn][scs][rn]( [rn][scs][rn]){2})((
53 [rn][scs][rn]){3})*)|\
54                 ( ?(A|a)( ?[rn][scs][rn] [rn] [0-1] [0-1]
55 [rn][scs][rn]))\

```

```

1                                     ( [rn][scs][rn] [rn] [0-1] [0-1]
2 [rn][scs][rn]))*\
3                                     ( ?(Z|z))\
4                                     )*/>
5 -->
6         <xs:pattern value="(F ?(0|1))?( ?(M|m)( ?((\-|\+)?(([\0-9]+\.[\0-
7 9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
8 9]+))((e|E)(\-|\+)?[0-9]+)?)))( ?(M|m)( ?((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
9 9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
10 |\+)?[0-9]+?)))( ?(L|l)( ?((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-
11 9]+)?( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)))( ((\-
12 |\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\0-
13 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?))*)|( ?(H|h|V|v)( ?((\-|\+)?(([\0-
14 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)))( ((\-|\+)?(([\0-9]+\.[\0-
15 9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?))*)|( ?(Q|q|S|s)( ?((\-|\+)?(([\0-9]+\.[\0-
16 9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
17 9]+))((e|E)(\-|\+)?[0-9]+?) ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-
18 9]+)?( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)))( ((\-
19 |\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-
20 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)){2})*|( ?(C|c)( ?((\-|\+)?(([\0-
21 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-9]+\.[\0-
22 9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)( ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
23 9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
24 |\+)?[0-9]+?))){2})( ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?))(
25 ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)){3})*|( ?(A|a)(
26 ?((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-
27 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?) ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
28 9]+))((e|E)(\-|\+)?[0-9]+?) [0-1] [0-1] ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
29 9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
30 |\+)?[0-9]+?)))( ((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?))( ?,
31 ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?) ((\-|\+)?(([\0-
32 9]+\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?) [0-1] [0-1] ((\-|\+)?(([\0-9]+\.[\0-
33 9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+?)( ?, ?)((\-|\+)?(([\0-9]+\.[\0-9]+)?)|(\.[\0-
34 9]+))((e|E)(\-|\+)?[0-9]+?))*)|( ?(Z|z)))*" />
35     </xs:restriction>
36 </xs:simpleType>
37
38 <!-- Abbreviated Geometry grammar for PatGeometry.Figures -->
39 <xs:simpleType name="ST_AbbrGeom">
40     <xs:restriction base="xs:string">
41         <xs:whiteSpace value="collapse" />
42 <!--
43         <xs:pattern value="( ?(M|m)( ?[rn][scs][rn]))*\
44         (\
45         ( ?(M|m)( ?[rn][scs][rn]))*\
46         ( ?(L|l)( ?[rn][scs][rn])( [rn][scs][rn]))*\
47         ( ?(H|h|V|v)( ?[rn])( [rn]))*\
48         ( ?(Q|q|S|s)( ?[rn][scs][rn] [rn][scs][rn]))((
49 [rn][scs][rn]){2})*|\
50         ( ?(C|c)( ?[rn][scs][rn]( [rn][scs][rn]){2})(
51 [rn][scs][rn]){3})*|\
52         ( ?(A|a)( ?[rn][scs][rn] [rn] [0-1] [0-1]
53 [rn][scs][rn]))\
54         ( [rn][scs][rn] [rn] [0-1] [0-1]
55 [rn][scs][rn]))*\

```

```

1         ( ?(Z|z))\
2         )*/>
3
4     -->
5         <xs:pattern value="( ?(M|m)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-
6 9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-
7 9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?))|((
8 ?(L|l)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)))*|(?
9 (H|h|V|v)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?))*)|(?
10 (Q|q|S|s)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)))*|(?
11 (C|c)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)))*|(?
12 (A|a)( ?((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)( ? , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[0-9]+)?)))*|(?
13 (Z|z)))*" />
14     </xs:restriction>
15 </xs:simpleType>
16
17 <!-- Image reference via Uri -->
18 <xs:simpleType name="ST_UriImage">
19     <xs:restriction base="xs:anyURI">
20         <xs:pattern value="([\^\{\}.*]*)" />
21     </xs:restriction>
22 </xs:simpleType>
23
24 <!-- Image reference via ColorConvertedBitmap -->
25 <xs:simpleType name="ST_CtxBmpImage">
26     <xs:restriction base="xs:string">
27         <xs:pattern value="\{ColorConvertedBitmap[\s]+[\S]+[\s]+[\S]+\}[\s]*" />
28     </xs:restriction>
29 </xs:simpleType>
30
31 <!-- Image reference via Uri or ColorConvertedBitmap -->
32 <xs:simpleType name="ST_UriCtxBmp">
33     <xs:union memberTypes="ST_UriImage ST_CtxBmpImage" />
34 </xs:simpleType>
35
36 <!-- Resource reference -->

```



```

1      <xs:simpleType name="ST_RscRef">
2          <xs:restriction base="xs:string">
3              <xs:pattern value="\{StaticResource[\s]+[\S]+\}[\s]*" />
4          </xs:restriction>
5      </xs:simpleType>
6
7      <!-- Resource reference OR Color -->
8      <xs:simpleType name="ST_RscRefColor">
9          <xs:union memberTypes="ST_Color ST_RscRef" />
10     </xs:simpleType>
11
12     <!-- Resource reference OR Compact Matrix-->
13     <xs:simpleType name="ST_RscRefMatrix">
14         <xs:union memberTypes="ST_Matrix ST_RscRef" />
15     </xs:simpleType>
16
17     <!-- Resource reference OR AbbrGeomF-->
18     <xs:simpleType name="ST_RscRefAbbrGeomF">
19         <xs:union memberTypes="ST_AbbrGeomF ST_RscRef" />
20     </xs:simpleType>
21
22     <!-- Sweep Direction enumeration -->
23     <xs:simpleType name="ST_SweepDirection">
24         <xs:restriction base="xs:string">
25             <xs:enumeration value="Clockwise" />
26             <xs:enumeration value="Counterclockwise" />
27         </xs:restriction>
28     </xs:simpleType>
29
30     <!-- Dash Cap enumeration -->
31     <xs:simpleType name="ST_DashCap">
32         <xs:restriction base="xs:string">
33             <xs:enumeration value="Flat" />
34             <xs:enumeration value="Round" />
35             <xs:enumeration value="Square" />
36             <xs:enumeration value="Triangle" />
37         </xs:restriction>
38     </xs:simpleType>
39
40     <!-- Line Cap enumeration -->
41     <xs:simpleType name="ST_LineCap">
42         <xs:restriction base="xs:string">
43             <xs:enumeration value="Flat" />
44             <xs:enumeration value="Round" />
45             <xs:enumeration value="Square" />
46             <xs:enumeration value="Triangle" />
47         </xs:restriction>
48     </xs:simpleType>
49
50     <!-- Line Join enumeration -->
51     <xs:simpleType name="ST_LineJoin">
52         <xs:restriction base="xs:string">
53             <xs:enumeration value="Miter" />
54             <xs:enumeration value="Bevel" />
55             <xs:enumeration value="Round" />

```

```

1      </xs:restriction>
2  </xs:simpleType>
3
4  <!-- Tile Mode enumeration -->
5  <xs:simpleType name="ST_TileMode">
6      <xs:restriction base="xs:string">
7          <xs:enumeration value="None" />
8          <xs:enumeration value="Tile" />
9          <xs:enumeration value="FlipX" />
10         <xs:enumeration value="FlipY" />
11         <xs:enumeration value="FlipXY" />
12     </xs:restriction>
13 </xs:simpleType>
14
15 <!-- Color Interpolation Mode enumeration -->
16 <xs:simpleType name="ST_ClrIntMode">
17     <xs:restriction base="xs:string">
18         <xs:enumeration value="ScRgbLinearInterpolation" />
19         <xs:enumeration value="SRgbLinearInterpolation" />
20     </xs:restriction>
21 </xs:simpleType>
22
23 <!-- SpreadMethod Mode enumeration -->
24 <xs:simpleType name="ST_SpreadMethod">
25     <xs:restriction base="xs:string">
26         <xs:enumeration value="Pad" />
27         <xs:enumeration value="Reflect" />
28         <xs:enumeration value="Repeat" />
29     </xs:restriction>
30 </xs:simpleType>
31
32 <!-- FillRule Mode enumeration -->
33 <xs:simpleType name="ST_FillRule">
34     <xs:restriction base="xs:string">
35         <xs:enumeration value="EvenOdd" />
36         <xs:enumeration value="NonZero" />
37     </xs:restriction>
38 </xs:simpleType>
39
40 <!-- Edge Mode enumeration -->
41 <xs:simpleType name="ST_EdgeMode">
42     <xs:restriction base="xs:string">
43         <xs:enumeration value="Aliased" />
44     </xs:restriction>
45 </xs:simpleType>
46
47 <!-- Style Simulation Enumeration -->
48 <xs:simpleType name="ST_StyleSimulations">
49     <xs:restriction base="xs:string">
50         <xs:enumeration value="None" />
51         <xs:enumeration value="ItalicSimulation" />
52         <xs:enumeration value="BoldSimulation" />
53         <xs:enumeration value="BoldItalicSimulation" />
54     </xs:restriction>
55 </xs:simpleType>

```

```
1
2     <!-- ViewUnits Enumeration -->
3     <xs:simpleType name="ST_ViewUnits">
4         <xs:restriction base="xs:string">
5             <xs:enumeration value="Absolute" />
6         </xs:restriction>
7     </xs:simpleType>
8
9     <!-- MappingMode Enumeration -->
10    <xs:simpleType name="ST_MappingMode">
11        <xs:restriction base="xs:string">
12            <xs:enumeration value="Absolute" />
13        </xs:restriction>
14    </xs:simpleType>
15 </xs:schema>
```


1 C. Resource Dictionary Key W3C Schema

2 The schema shown below is also provided in electronic form as a file named RDKey.xsd, which
3 is contained in an accompanying zip archive named "XPS WC3 Schemas.zip". If discrepancies
4 exist between the representation as published below and the corresponding electronic version,
5 the published version below is the definitive version

```
6
7 <?xml version="1.0" encoding="utf-8"?>
8 <xs:schema
9   targetNamespace="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-key"
10  xmlns="http://schemas.microsoft.com/xps/2005/06/resourcedictionary-key"
11  xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
12  blockDefault="#all">
13
14   <xs:attribute name="Key">
15     <xs:simpleType>
16       <xs:restriction base="xs:string">
17         <!-- A Key (pattern restriction according to XPS spec) -->
18         <xs:pattern
19   value="(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|_)(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{Nl}|\p{Mn}
20 |\p{Mc}|\p{Nd}|_)*" />
21       </xs:restriction>
22     </xs:simpleType>
23   </xs:attribute>
24
25 </xs:schema>
```


1 D. Document Structure W3C Schema

2 The schema shown below is also provided in electronic form as a file named DocStructure.xsd,
3 which is contained in an accompanying zip archive named "XPS WC3 Schemas.zip". If
4 discrepancies exist between the representation as published below and the corresponding
5 electronic version, the published version below is the definitive version

```
6
7 <?xml version="1.0" encoding="UTF-8"?>
8 <xs:schema targetNamespace="http://schemas.microsoft.com/xps/2005/06/documentstructure"
9 xmlns="http://schemas.microsoft.com/xps/2005/06/documentstructure"
10 xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
11 blockDefault="#all">
12
13     <xs:import namespace="http://www.w3.org/XML/1998/namespace" />
14
15     <!-- =====DocumentStructure Part===== -->
16     <!-- Complex Types -->
17     <xs:complexType name="CT_DocumentStructure">
18         <xs:sequence>
19             <xs:element ref="DocumentStructure.Outline" minOccurs="0" />
20             <xs:element ref="Story" minOccurs="0" maxOccurs="unbounded" />
21         </xs:sequence>
22     </xs:complexType>
23     <xs:complexType name="CT_CP_Outline">
24         <xs:sequence>
25             <xs:element ref="DocumentOutline" />
26         </xs:sequence>
27     </xs:complexType>
28     <xs:complexType name="CT_DocumentOutline">
29         <xs:sequence>
30             <xs:element ref="OutlineEntry" maxOccurs="unbounded" />
31         </xs:sequence>
32         <xs:attributeGroup ref="AG_DocumentOutline" />
33     </xs:complexType>
34     <xs:complexType name="CT_OutlineEntry">
35         <xs:attributeGroup ref="AG_OutlineEntry" />
36     </xs:complexType>
37     <xs:complexType name="CT_Story">
38         <xs:sequence>
39             <xs:element ref="StoryFragmentReference" maxOccurs="unbounded" />
40         </xs:sequence>
41         <xs:attributeGroup ref="AG_Story" />
42     </xs:complexType>
43     <xs:complexType name="CT_StoryFragmentReference">
44         <xs:attributeGroup ref="AG_StoryFragmentReference" />
45     </xs:complexType>
46     <!-- Simple Types -->
47     <!-- A Name (ID with pattern restriction according to XPS spec) -->
48     <xs:simpleType name="ST_Name">
49         <xs:restriction base="xs:string">
```

```

1         <xs:pattern
2 value="(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl})(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl}|\p{Mn}|\
3 p{Mc}|\p{Nd}|\p{Lm}|_)*" />
4         </xs:restriction>
5     </xs:simpleType>
6     <!-- A Unique Name (ID with pattern restriction according to XPS spec) -->
7     <xs:simpleType name="ST_NameUnique">
8         <xs:restriction base="xs:ID">
9             <xs:pattern
10 value="(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl})(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl}|\p{Mn}|\
11 p{Mc}|\p{Nd}|\p{Lm}|_)*" />
12             </xs:restriction>
13         </xs:simpleType>
14         <!-- integer greater than or equal to 1 inclusive -->
15         <xs:simpleType name="ST_IntGEOne">
16             <xs:restriction base="xs:int">
17                 <xs:minInclusive value="1" />
18             </xs:restriction>
19         </xs:simpleType>
20         <!-- Elements -->
21         <xs:element name="DocumentStructure" type="CT_DocumentStructure">
22             </xs:element>
23         <xs:element name="DocumentStructure.Outline" type="CT_CP_Outline">
24             </xs:element>
25         <xs:element name="DocumentOutline" type="CT_DocumentOutline">
26             </xs:element>
27         <xs:element name="OutlineEntry" type="CT_OutlineEntry">
28             </xs:element>
29         <xs:element name="Story" type="CT_Story">
30             </xs:element>
31         <xs:element name="StoryFragmentReference" type="CT_StoryFragmentReference">
32             </xs:element>
33         <!-- Attribute Groups -->
34         <xs:attributeGroup name="AG_DocumentOutline">
35             <xs:attribute ref="xml:lang" use="required">
36                 </xs:attribute>
37         </xs:attributeGroup>
38         <xs:attributeGroup name="AG_OutlineEntry">
39             <xs:attribute name="OutlineLevel" type="ST_IntGEOne" use="optional"
40 default="1">
41                 </xs:attribute>
42             <xs:attribute name="OutlineTarget" type="xs:anyURI" use="required">
43                 </xs:attribute>
44             <xs:attribute name="Description" type="xs:string" use="required">
45                 </xs:attribute>
46             <xs:attribute ref="xml:lang" use="optional">
47                 </xs:attribute>
48         </xs:attributeGroup>
49         <xs:attributeGroup name="AG_Story">
50             <xs:attribute name="StoryName" type="xs:string" use="required">
51                 </xs:attribute>
52         </xs:attributeGroup>
53         <xs:attributeGroup name="AG_StoryFragmentReference">
54             <xs:attribute name="FragmentName" type="xs:string" use="optional">
55                 </xs:attribute>

```



```

1         <xs:attribute name="Page" type="ST_IntGEOne" use="required">
2             </xs:attribute>
3     </xs:attributeGroup>
4     <!-- =====StoryFragments Part===== -->
5     <!-- Complex Types -->
6     <xs:complexType name="CT_StoryFragments">
7         <xs:sequence>
8             <xs:element ref="StoryFragment" maxOccurs="unbounded" />
9         </xs:sequence>
10    </xs:complexType>
11    <xs:complexType name="CT_StoryFragment">
12        <xs:sequence>
13            <xs:element ref="StoryBreak" minOccurs="0" />
14            <xs:choice maxOccurs="unbounded">
15                <xs:element ref="SectionStructure" />
16                <xs:element ref="ParagraphStructure" />
17                <xs:element ref="ListStructure" />
18                <xs:element ref="TableStructure" />
19                <xs:element ref="FigureStructure" />
20            </xs:choice>
21            <xs:element ref="StoryBreak" minOccurs="0" />
22        </xs:sequence>
23        <xs:attributeGroup ref="AG_StoryFragment" />
24    </xs:complexType>
25    <xs:complexType name="CT_Break">
26        </xs:complexType>
27    <xs:complexType name="CT_Section">
28        <xs:choice maxOccurs="unbounded">
29            <xs:element ref="ParagraphStructure" />
30            <xs:element ref="ListStructure" />
31            <xs:element ref="TableStructure" />
32            <xs:element ref="FigureStructure" />
33        </xs:choice>
34    </xs:complexType>
35    <xs:complexType name="CT_Paragraph">
36        <xs:choice minOccurs="0" maxOccurs="unbounded">
37            <xs:element ref="NamedElement" />
38        </xs:choice>
39    </xs:complexType>
40    <xs:complexType name="CT_Table">
41        <xs:choice maxOccurs="unbounded">
42            <xs:element ref="TableRowGroupStructure" />
43        </xs:choice>
44    </xs:complexType>
45    <xs:complexType name="CT_TableRowGroup">
46        <xs:choice maxOccurs="unbounded">
47            <xs:element ref="TableRowStructure" />
48        </xs:choice>
49    </xs:complexType>
50    <xs:complexType name="CT_TableRow">
51        <xs:choice maxOccurs="unbounded">
52            <xs:element ref="TableCellStructure" />
53        </xs:choice>
54    </xs:complexType>
55    <xs:complexType name="CT_TableCell">

```

```

1         <xs:choice minOccurs="0" maxOccurs="unbounded">
2             <xs:element ref="ParagraphStructure" />
3             <xs:element ref="ListStructure" />
4             <xs:element ref="TableStructure" />
5             <xs:element ref="FigureStructure" />
6         </xs:choice>
7         <xs:attributeGroup ref="AG_TableCell" />
8     </xs:complexType>
9     <xs:complexType name="CT_List">
10        <xs:choice maxOccurs="unbounded">
11            <xs:element ref="ListItemStructure" />
12        </xs:choice>
13    </xs:complexType>
14    <xs:complexType name="CT_ListItem">
15        <xs:choice minOccurs="0" maxOccurs="unbounded">
16            <xs:element ref="ParagraphStructure" />
17            <xs:element ref="ListStructure" />
18            <xs:element ref="TableStructure" />
19            <xs:element ref="FigureStructure" />
20        </xs:choice>
21        <xs:attributeGroup ref="AG_ListItem" />
22    </xs:complexType>
23    <xs:complexType name="CT_Figure">
24        <xs:choice minOccurs="0" maxOccurs="unbounded">
25            <xs:element ref="NamedElement" />
26        </xs:choice>
27    </xs:complexType>
28    <xs:complexType name="CT_NamedElement">
29        <xs:attributeGroup ref="AG_NamedElement" />
30    </xs:complexType>
31    <!-- Simple Types -->
32    <!-- FragmentType enumeration -->
33    <xs:simpleType name="ST_FragmentType">
34        <xs:restriction base="xs:string">
35            <xs:enumeration value="Content" />
36            <xs:enumeration value="Header" />
37            <xs:enumeration value="Footer" />
38        </xs:restriction>
39    </xs:simpleType>
40    <xs:simpleType name="ST_Location">
41        <xs:restriction base="xs:string">
42            <xs:pattern value="([0-9][0-9]*)(\,[0-9][0-9]*)*" />
43        </xs:restriction>
44    </xs:simpleType>
45    <xs:simpleType name="ST_TableSpan">
46        <xs:restriction base="xs:int">
47            <xs:minInclusive value="1" />
48        </xs:restriction>
49    </xs:simpleType>
50    <xs:simpleType name="ST_ElementIndex">
51        <xs:restriction base="xs:int">
52            <xs:minInclusive value="0" />
53        </xs:restriction>
54    </xs:simpleType>
55    <!-- Elements -->

```

```

1      <xs:element name="StoryFragments" type="CT_StoryFragments">
2          </xs:element>
3      <xs:element name="StoryFragment" type="CT_StoryFragment">
4          </xs:element>
5      <xs:element name="StoryBreak" type="CT_Break">
6          </xs:element>
7      <xs:element name="SectionStructure" type="CT_Section">
8          </xs:element>
9      <xs:element name="ParagraphStructure" type="CT_Paragraph">
10         </xs:element>
11     <xs:element name="TableStructure" type="CT_Table">
12         </xs:element>
13     <xs:element name="TableRowGroupStructure" type="CT_TableRowGroup">
14         </xs:element>
15     <xs:element name="TableRowStructure" type="CT_TableRow">
16         </xs:element>
17     <xs:element name="TableCellStructure" type="CT_TableCell">
18         </xs:element>
19     <xs:element name="ListStructure" type="CT_List">
20         </xs:element>
21     <xs:element name="ListItemStructure" type="CT_ListItem">
22         </xs:element>
23     <xs:element name="FigureStructure" type="CT_Figure">
24         </xs:element>
25     <xs:element name="NamedElement" type="CT_NamedElement">
26         </xs:element>
27     <!-- Attribute Groups -->
28     <xs:attributeGroup name="AG_StoryFragment">
29         <xs:attribute name="StoryName" type="xs:string" use="optional">
30             </xs:attribute>
31         <xs:attribute name="FragmentName" type="xs:string" use="optional">
32             </xs:attribute>
33         <xs:attribute name="FragmentType" type="ST_FragmentType" use="required">
34             </xs:attribute>
35     </xs:attributeGroup>
36     <xs:attributeGroup name="AG_TableCell">
37         <xs:attribute name="RowSpan" type="ST_TableSpan" use="optional"
38 default="1">
39             </xs:attribute>
40         <xs:attribute name="ColumnSpan" type="ST_TableSpan" use="optional"
41 default="1">
42             </xs:attribute>
43     </xs:attributeGroup>
44     <xs:attributeGroup name="AG_ListItem">
45         <xs:attribute name="Marker" type="ST_NameUnique" use="optional">
46             </xs:attribute>
47     </xs:attributeGroup>
48     <xs:attributeGroup name="AG_NamedElement">
49         <xs:attribute name="NameReference" type="ST_Name" use="required">
50             </xs:attribute>
51     </xs:attributeGroup>
52 </xs:schema>
53
54

```


1 E. Discard Control W3C Schema

2 The schema shown below is also provided in electronic form as a file named DiscardControl.xsd,
3 which is contained in an accompanying zip archive named "XPS WC3 Schemas.zip". If
4 discrepancies exist between the representation as published below and the corresponding
5 electronic version, the published version below is the definitive version

```
6
7 <?xml version="1.0" encoding="UTF-8"?>
8 <xs:schema targetNamespace="http://schemas.microsoft.com/xps/2005/06/discard-control"
9 xmlns="http://schemas.microsoft.com/xps/2005/06/discard-control"
10 xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
11 blockDefault="#all">
12
13     <xs:complexType name="CT_DiscardControl">
14         <xs:sequence>
15             <xs:element ref="Discard" minOccurs="0" maxOccurs="unbounded" />
16         </xs:sequence>
17     </xs:complexType>
18
19     <xs:complexType name="CT_Discard">
20         <xs:attribute name="SentinelPage" type="xs:anyURI" use="required">
21             </xs:attribute>
22         <xs:attribute name="Target" type="xs:anyURI" use="required">
23             </xs:attribute>
24     </xs:complexType>
25
26     <xs:element name="DiscardControl" type="CT_DiscardControl">
27         </xs:element>
28
29     <xs:element name="Discard" type="CT_Discard">
30         </xs:element>
31
32 </xs:schema>
```


1 F. Abbreviated Geometry Syntax Algorithm

2 A path geometry specified using the abbreviated geometry syntax (see §11.2.3) is equivalent
 3 to a path specified using a path geometry. The following algorithm describes how the
 4 abbreviated path syntax can be transformed into a path geometry containing path figures that,
 5 in turn, contain various segments.

6 This algorithm assumes that the presented string is well-formed according to the markup
 7 schema. Whitespace skipping is assumed without being explicitly spelled out in the algorithm.

```

8
9   Let CURRENTPOINT = 0,0
10  Create a new PathGeometry PG
11  PG.FillRule = EvenOdd
12  Let CURRENTPATHFIGURE = undefined
13
14  Read input character CH
15
16  if ( CH == 'F' )
17  {
18      Read input character CH
19      if ( CH == '0' )
20      {
21          PG.FillRule = EvenOdd
22      }
23      else
24      {
25          PG.FillRule = NonZero
26      }
27  }
28  else
29  {
30      GOTO label_first
31  }
32
33  label_repeat:
34  Read input character CH
35
36  label_first:
37
38  if ( CH == 'm' )
39  {
40      Read relative coordinate pair DX,DY
41      Let CURRENTPOINT.X = CURRENTPOINT.X + DX
42      Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
43      Create a new PathFigure CURRENTPATHFIGURE and add to PG
44      Let attribute CURRENTPATHFIGURE.StartPoint = CURRENTPOINT
45  }
46  else if ( CH == 'M' )
47  {
48      Read coordinate pair X,Y
49      Let CURRENTPOINT.X = X
50      Let CURRENTPOINT.Y = Y
51      Create a new PathFigure CURRENTPATHFIGURE and add to PG
52      Let attribute CURRENTPATHFIGURE.StartPoint = CURRENTPOINT
53  }
54  else if ( CH == 'l' )
55  {
56      Create new PolyLineSegment S
57      Add S to CURRENTPATHFIGURE
58  label_1:

```

```

1         Read relative coordinate pair DX,DY
2         Let CURRENTPOINT.X = CURRENTPOINT.X + DX
3         Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
4         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
5         if ( next character is not a letter )
6         {
7             GOTO label_1
8         }
9     }
10    else if ( CH == 'L' )
11    {
12        Create new PolyLineSegment S
13        Add S to CURRENTPATHFIGURE
14    label_2:
15        Read coordinate pair X,Y
16        Let CURRENTPOINT.X = X
17        Let CURRENTPOINT.Y = Y
18        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
19        if ( next character is not a letter )
20        {
21            GOTO label_2
22        }
23    }
24    else if ( CH == 'h' )
25    {
26        Create new PolyLineSegment S
27        Add S to CURRENTPATHFIGURE
28    label_3:
29        Read relative coordinate value DX
30        Let CURRENTPOINT.X = CURRENTPOINT.X + DX
31        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
32        if ( next character is not a letter )
33        {
34            GOTO label_3
35        }
36    }
37    else if ( CH == 'H' )
38    {
39        Create new PolyLineSegment S
40        Add S to CURRENTPATHFIGURE
41    label_4:
42        Read coordinate value X
43        Let CURRENTPOINT.X = X
44        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
45        if ( next character is not a letter )
46        {
47            GOTO label_4
48        }
49    }
50    else if ( CH == 'v' )
51    {
52        Create new PolyLineSegment S
53        Add S to CURRENTPATHFIGURE
54    label_5:
55        Read relative coordinate value DY
56        Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
57        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
58        if ( next character is not a letter )
59        {
60            GOTO label_5
61        }
62    }
63    else if ( CH == 'V' )
64    {
65        Create new PolyLineSegment S
66        Add S to CURRENTPATHFIGURE
67    label_6:
68        Read coordinate value Y
69        Let CURRENTPOINT.Y = Y

```



```

1          Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
2          if ( next character is not a letter )
3          {
4              GOTO label_6
5          }
6      }
7      else if ( CH == 'c' )
8      {
9          Create new PolyBezierSegment S
10         Add S to CURRENTPATHFIGURE
11     label_7:
12         Read relative coordinate pair DX,DY
13         Let POINT.X = CURRENTPOINT.X + DX
14         Let POINT.Y = CURRENTPOINT.Y + DY
15         Add POINT.X, POINT.Y to end of S.Points attribute list
16         Read coordinate pair DX,DY
17         Let POINT.X = CURRENTPOINT.X + DX
18         Let POINT.Y = CURRENTPOINT.Y + DY
19         Add POINT.X, POINT.Y to end of S.Points attribute list
20         Read coordinate pair DX,DY
21         Let CURRENTPOINT.X = CURRENTPOINT.X + DX
22         Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
23         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
24         if ( next character is not a letter )
25         {
26             GOTO label_7
27         }
28     }
29     else if ( CH == 'C' )
30     {
31         Create new PolyBezierSegment S
32         Add S to CURRENTPATHFIGURE
33     label_8:
34         Read coordinate pair X,Y
35         Let POINT.X = X
36         Let POINT.Y = Y
37         Add POINT.X, POINT.Y to end of S.Points attribute list
38         Read coordinate pair X,Y
39         Let POINT.X = X
40         Let POINT.Y = Y
41         Add POINT.X, POINT.Y to end of S.Points attribute list
42         Read coordinate pair X,Y
43         Let CURRENTPOINT.X = X
44         Let CURRENTPOINT.Y = Y
45         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
46         if ( next character is not a letter )
47         {
48             GOTO label_8
49         }
50     }
51     else if ( CH == 'q' )
52     {
53         Create new PolyQuadraticBezierSegment S
54         Add S to CURRENTPATHFIGURE
55     label_9:
56         Read relative coordinate pair DX,DY
57         Let POINT.X = CURRENTPOINT.X + DX
58         Let POINT.Y = CURRENTPOINT.Y + DY
59         Add POINT.X, POINT.Y to end of S.Points attribute list
60         Read relative coordinate pair DX,DY
61         Let CURRENTPOINT.X = CURRENTPOINT.X + DX
62         Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
63         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
64         if ( next character is not a letter )
65         {
66             GOTO label_9
67         }
68     }

```

```

1      else ( if CH == 'Q' )
2      {
3          Create new PolyQuadraticBezierSegment S
4          Add S to CURRENTPATHFIGURE
5      label_10:
6          Read coordinate pair X,Y
7          Let POINT.X = X
8          Let POINT.Y = Y
9          Add POINT.X, POINT.Y to end of S.Points attribute list
10         Read coordinate pair X,Y
11         Let CURRENTPOINT.X = X
12         Let CURRENTPOINT.Y = Y
13         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
14         if ( next character is not a letter )
15         {
16             GOTO label_10
17         }
18     }
19     else if ( CH == 's' )
20     {
21         Create new PolyBezierSegment S
22         Add S to CURRENTPATHFIGURE
23     label_11:
24         if ( S.Points is non-empty )
25         {
26             Let LASTCTRLPOINT = Point before last point in S.Points
27             Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
28             Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
29         }
30         else if ( segment before CURRENTPATHSEGMENT is a PolyBezierSegment )
31         {
32             Let LASTCTRLPOINT = Point before last point in previous PolyBezierSegment
33             Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
34             Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
35         }
36         else
37         {
38             Let POINT = CURRENTPOINT
39         }
40
41         Add POINT.X, POINT.Y to end of S.Points attribute list
42         Read relative coordinate pair DX,DY
43         Let POINT.X = CURRENTPOINT.X + DX
44         Let POINT.Y = CURRENTPOINT.Y + DY
45         Add POINT.X, POINT.Y to end of S.Points attribute list
46         Read relative coordinate pair DX,DY
47         Let CURRENTPOINT.X = CURRENTPOINT.X + DX
48         Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
49         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
50         if ( next character is not a letter )
51         {
52             GOTO label_11
53         }
54     }
55     else if ( CH == 'S' )
56     {
57         Create new PolyBezierSegment S
58         Add S to CURRENTPATHFIGURE
59     label_12:
60         if ( S.Points is non-empty )
61         {
62             Let LASTCTRLPOINT = Point before last point in S.Points
63             Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
64             Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
65         }
66         else if ( segment before CURRENTPATHSEGMENT is a PolyBezierSegment )
67         {
68             Let LASTCTRLPOINT = S.Point before last point in previous PolyBezierSegment
69             Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X

```

```

1           Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
2       }
3   else
4       {
5           Let POINT = CURRENTPOINT
6       }
7
8       Add POINT.X, POINT.Y to end of S.Points attribute list
9       Read coordinate pair X,Y
10      Let POINT.X = X
11      Let POINT.Y = Y
12      Add POINT.X, POINT.Y to end of S.Points attribute list
13      Read coordinate pair X,Y
14      Let CURRENTPOINT.X = X
15      Let CURRENTPOINT.Y = Y
16      Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
17      if ( next character is not a letter )
18          {
19              GOTO label_12
20          }
21      }
22  else if ( CH == 'a' )
23      {
24  label_13:
25      Create new ArcSegment S
26      Add S to CURRENTPATHFIGURE
27      Read Radius Pair RX,RY
28      Read Rotation ROT
29      Read integer FLAG1
30      Read integer FLAG2
31      Read relative coordinate pair DX,DY
32      Let CURRENTPOINT.X = CURRENTPOINT.X + DX
33      Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
34      Let S.Point = CURRENTPOINT.X,CURRENTPOINT.Y
35      Let S.IsLargeArc = (FLAG1 == 1 ? true : false)
36      Let S.SweepDirection = (FLAG2 == 1 ? Clockwise : Counterclockwise)
37      Let S.RotationAngle = ROT
38      Let S.Size = RX, RY
39      if ( next character is not a letter )
40          {
41              GOTO label_13
42          }
43      }
44  else if ( CH == 'A' )
45      {
46  label_14:
47      Create new ArcSegment S
48      Add S to CURRENTPATHFIGURE
49      Read Radius Pair RX,RY
50      Read Rotation ROT
51      Read integer FLAG1
52      Read integer FLAG2
53      Read coordinate pair X,Y
54      Let CURRENTPOINT.X = X
55      Let CURRENTPOINT.Y = Y
56      Let S.Point = CURRENTPOINT.X,CURRENTPOINT.Y
57      Let S.IsLargeArc = (FLAG1 == 1 ? true : false)
58      Let S.SweepDirection = (FLAG2 == 1 ? Clockwise : Counterclockwise)
59      Let S.RotationAngle = ROT
60      Let S.Size = RX, RY
61      if ( next character is not a letter )
62          {
63              GOTO label_14
64          }
65      }
66  else if ( CH == 'z' or CH == 'Z' )
67      {
68      Let attribute CURRENTPATHFIGURE.IsClosed = true
69      Let CURRENTPOINT = First point of first segment of CURRENTPATHFIGURE

```

```
1           Let CURRENTPATHFIGURE = undefined
2       }
3   /* This case can not occur, because the input is assumed to be well-formed according to the markup
4   schema
5   else
6   {
7       ERROR: Invalid input character
8   }
9   */
10
11   if ( End of input reached )
12   {
13       Terminate algorithm, return PG
14   }
15   else
16   {
17       GOTO label_repeat
18   }
19
```

1 G. scRGB Gamut Boundary Definition

```

2 <?xml version="1.0" encoding="utf-8" ?>
3 <cdm:ColorDeviceModel
4     ID="http://schemas.microsoft.com/windows/2005/02/color/wcsRGB.cdmp"
5     xmlns:cdm="http://schemas.microsoft.com/windows/2005/02/color/
6         ColorDeviceModel"
7     xmlns:wcs="http://schemas.microsoft.com/windows/2005/02/color/
8         WcsCommonProfileTypes"
9     xmlns:xs='http://www.w3.org/2001/XMLSchema-instance'>
10
11     <cdm:ProfileName>
12         <wcs:Text xml:lang="en-US">wcsRGB virtual device model profile</wcs:Text>
13     </cdm:ProfileName>
14     <cdm:Description>
15         <wcs:Text xml:lang="en-US">wcsRGB virtual s2.13 device model system
16 profile</wcs:Text>
17     </cdm:Description>
18     <cdm:Author>
19         <wcs:Text xml:lang="en-US">Microsoft Corporation</wcs:Text>
20     </cdm:Author>
21
22     <cdm:MeasurementConditions>
23         <cdm:ColorSpace>CIEXYZ</cdm:ColorSpace>
24         <cdm:WhitePointName>D65</cdm:WhitePointName>
25     </cdm:MeasurementConditions>
26     <cdm:SelfLuminous>true</cdm:SelfLuminous>
27     <cdm:MaxColorant>4.0</cdm:MaxColorant>
28     <cdm:MinColorant>-4.0</cdm:MinColorant>
29
30     <cdm:RGBVirtualDevice>
31         <cdm:MeasurementData TimeStamp="2005-02-09T22:00:00">
32             <cdm:WhitePrimary X="95.05" Y="100.00" Z="108.90" />
33             <cdm:RedPrimary X="41.24" Y="21.26" Z="1.93" />
34             <cdm:GreenPrimary X="35.76" Y="71.52" Z="11.92" />
35             <cdm:BluePrimary X="18.05" Y="7.22" Z="95.05" />
36             <cdm:BlackPrimary X="0" Y="0" Z="0" />
37             <cdm:Gamma value="1.0" />
38             <cdm:GamutBoundarySamples>
39                 <cdm:RGB R="0.166433" G="-0.016114" B="0.027918" />
40                 <cdm:RGB R="0.257134" G="-0.026542" B="0.041950" />
41                 <cdm:RGB R="0.373396" G="-0.039395" B="0.060310" />
42                 <cdm:RGB R="0.516590" G="-0.054207" B="0.083665" />
43                 <cdm:RGB R="0.161885" G="-0.013701" B="0.015224" />
44                 <cdm:RGB R="0.254873" G="-0.024242" B="0.022314" />
45                 <cdm:RGB R="0.370591" G="-0.036207" B="0.031873" />
46                 <cdm:RGB R="0.645317" G="-0.053422" B="0.061566" />
47                 <cdm:RGB R="0.246052" G="-0.020570" B="0.009804" />
48                 <cdm:RGB R="0.366951" G="-0.033595" B="0.013350" />
49                 <cdm:RGB R="0.513086" G="-0.047777" B="0.018342" />
50                 <cdm:RGB R="0.348159" G="-0.027092" B="0.002697" />
51                 <cdm:RGB R="0.501415" G="-0.043030" B="0.003038" />

```

```

1      <cdm:RGB R="0.204158" G="-0.007054" B="-0.001921" />
2      <cdm:RGB R="0.102843" G="0.005460" B="-0.002574" />
3      <cdm:RGB R="0.161680" G="0.005828" B="-0.003751" />
4      <cdm:RGB R="0.241218" G="0.005609" B="-0.005263" />
5      <cdm:RGB R="0.345298" G="0.004589" B="-0.007162" />
6      <cdm:RGB R="1.381166" G="0.758736" B="0.088389" />
7      <cdm:RGB R="0.747224" G="0.490147" B="-0.065045" />
8      <cdm:RGB R="0.892023" G="0.585517" B="-0.076908" />
9      <cdm:RGB R="1.054700" G="0.693668" B="-0.087994" />
10     <cdm:RGB R="1.237403" G="0.815408" B="-0.099715" />
11     <cdm:RGB R="0.429786" G="0.463528" B="-0.056073" />
12     <cdm:RGB R="0.109329" G="0.215829" B="-0.023359" />
13     <cdm:RGB R="0.170974" G="0.342970" B="-0.037798" />
14     <cdm:RGB R="0.032781" G="0.182702" B="-0.017604" />
15     <cdm:RGB R="0.040703" G="0.236387" B="-0.023182" />
16     <cdm:RGB R="0.052010" G="0.299807" B="-0.029289" />
17     <cdm:RGB R="-0.015232" G="0.150695" B="-0.012907" />
18     <cdm:RGB R="-0.020689" G="0.198705" B="-0.017162" />
19     <cdm:RGB R="-0.086126" G="0.133080" B="0.007658" />
20     <cdm:RGB R="-0.178409" G="0.235796" B="0.126003" />
21     <cdm:RGB R="0.000000" G="0.000000" B="0.000000" />
22     <cdm:RGB R="-0.150496" G="0.178902" B="0.156523" />
23     <cdm:RGB R="-0.193313" G="0.234293" B="0.204426" />
24     <cdm:RGB R="-0.202798" G="0.230449" B="0.290886" />
25     <cdm:RGB R="-0.211558" G="0.225780" B="0.384316" />
26     <cdm:RGB R="-0.249008" G="0.286162" B="0.470316" />
27     <cdm:RGB R="1.024267" G="0.995126" B="0.956518" />
28     <cdm:RGB R="-0.211156" G="0.218574" B="0.473918" />
29     <cdm:RGB R="-0.209022" G="0.210461" B="0.567755" />
30     <cdm:RGB R="0.368209" G="0.814614" B="1.020972" />
31     <cdm:RGB R="-0.194449" G="0.199547" B="0.648518" />
32     <cdm:RGB R="0.099507" G="0.602350" B="1.001310" />
33     <cdm:RGB R="0.447496" G="0.790515" B="1.022822" />
34     <cdm:RGB R="-0.123046" G="0.108061" B="0.470343" />
35     <cdm:RGB R="-0.150585" G="0.144010" B="0.596962" />
36     <cdm:RGB R="0.185831" G="0.575754" B="1.007060" />
37     <cdm:RGB R="0.331216" G="0.668936" B="1.023494" />
38     <cdm:RGB R="0.410196" G="0.645807" B="1.014063" />
39     <cdm:RGB R="0.574938" G="0.752116" B="1.021593" />
40     <cdm:RGB R="0.483140" G="0.624458" B="1.005310" />
41     <cdm:RGB R="0.635447" G="0.734762" B="1.010061" />
42     <cdm:RGB R="-0.052205" G="0.035138" B="0.436365" />
43     <cdm:RGB R="1.024267" G="0.995126" B="0.956518" />
44     <cdm:RGB R="0.028844" G="-0.005277" B="0.123163" />
45     <cdm:RGB R="0.049636" G="-0.010904" B="0.228673" />
46     <cdm:RGB R="0.080157" G="-0.012435" B="0.135751" />
47     <cdm:RGB R="0.127487" G="-0.020154" B="0.216960" />
48     <cdm:RGB R="0.190206" G="-0.029844" B="0.323182" />
49     <cdm:RGB R="0.097580" G="-0.013313" B="0.083401" />
50     <cdm:RGB R="0.155931" G="-0.021750" B="0.133422" />
51     <cdm:RGB R="0.230038" G="-0.031267" B="0.196426" />
52     <cdm:RGB R="0.326426" G="-0.043430" B="0.278289" />
53     <cdm:RGB R="0.426090" G="-0.050043" B="0.360668" />
54     <cdm:RGB R="0.098869" G="-0.010753" B="0.048113" />
55     <cdm:RGB R="0.164545" G="-0.019812" B="0.079230" />

```

```

1      <cdm:RGB R="0.249557" G="-0.030741" B="0.119834" />
2      <cdm:RGB R="0.357564" G="-0.043900" B="0.171719" />
3      <cdm:RGB R="0.495437" G="-0.060672" B="0.237960" />
4      <cdm:RGB R="0.619457" G="-0.065550" B="0.301872" />
5      <cdm:RGB R="0.000000" G="0.000000" B="0.000000" />
6      <cdm:RGB R="0.169101" G="-0.018572" B="0.047904" />
7      <cdm:RGB R="0.255247" G="-0.028467" B="0.071995" />
8      <cdm:RGB R="0.368932" G="-0.041653" B="0.103698" />
9      <cdm:RGB R="0.508606" G="-0.056762" B="0.143412" />
10     <cdm:RGB R="0.637110" G="-0.061275" B="0.186481" />
11     <cdm:RGB R="1.249149" G="-0.024436" B="-0.002194" />
12     <cdm:RGB R="0.000475" G="-0.001355" B="0.043728" />
13     <cdm:RGB R="1.257972" G="-0.026125" B="0.049916" />
14     <cdm:RGB R="1.264195" G="-0.027432" B="0.104713" />
15     <cdm:RGB R="0.003291" G="-0.009926" B="0.365422" />
16     <cdm:RGB R="1.287764" G="-0.033804" B="0.414704" />
17     <cdm:RGB R="0.004536" G="-0.013745" B="0.512430" />
18     <cdm:RGB R="1.297183" G="-0.037024" B="0.573653" />
19     <cdm:RGB R="1.307374" G="-0.040929" B="0.763754" />
20     <cdm:RGB R="0.007818" G="-0.023805" B="0.902837" />
21     <cdm:RGB R="-0.001325" G="0.002612" B="-0.000199" />
22     <cdm:RGB R="-0.007337" G="0.015263" B="-0.001058" />
23     <cdm:RGB R="-0.005430" G="-0.000728" B="0.906022" />
24     <cdm:RGB R="-0.020327" G="0.043021" B="-0.002887" />
25     <cdm:RGB R="-0.042091" G="0.089818" B="-0.005932" />
26     <cdm:RGB R="-0.117963" G="0.253649" B="-0.016500" />
27     <cdm:RGB R="-0.141327" G="0.262090" B="0.943868" />
28     <cdm:RGB R="-0.174730" G="0.376455" B="-0.024386" />
29     <cdm:RGB R="-0.204641" G="0.390245" B="0.954021" />
30     <cdm:RGB R="-0.245665" G="0.530007" B="-0.034229" />
31     <cdm:RGB R="-0.282073" G="0.548898" B="0.962351" />
32     <cdm:RGB R="1.150086" G="0.568927" B="1.001779" />
33     <cdm:RGB R="-0.374737" G="0.740509" B="0.968358" />
34     <cdm:RGB R="1.083883" G="0.766726" B="1.001175" />
35     <cdm:RGB R="1.002593" G="0.985026" B="-0.073045" />
36     <cdm:RGB R="-0.434478" G="0.938894" B="-0.060390" />
37     <cdm:RGB R="0.000055" G="-0.000136" B="0.003135" />
38     <cdm:RGB R="0.000203" G="-0.000555" B="0.016188" />
39     <cdm:RGB R="0.000896" G="-0.002619" B="0.089419" />
40     <cdm:RGB R="0.001492" G="-0.004425" B="0.156418" />
41     <cdm:RGB R="0.002283" G="-0.006839" B="0.247555" />
42     <cdm:RGB R="0.006038" G="-0.018353" B="0.690852" />
43     <cdm:RGB R="0.007818" G="-0.023805" B="0.902837" />
44     <cdm:RGB R="-0.483692" G="0.967403" B="0.971726" />
45     <cdm:RGB R="0.003728" G="-0.000194" B="0.002807" />
46     <cdm:RGB R="0.004140" G="-0.000555" B="0.015782" />
47     <cdm:RGB R="0.004738" G="-0.001328" B="0.043372" />
48     <cdm:RGB R="0.005489" G="-0.002582" B="0.089155" />
49     <cdm:RGB R="0.006397" G="-0.004383" B="0.156254" />
50     <cdm:RGB R="0.007479" G="-0.006795" B="0.247483" />
51     <cdm:RGB R="0.008758" G="-0.009881" B="0.365433" />
52     <cdm:RGB R="0.010255" G="-0.013700" B="0.512516" />
53     <cdm:RGB R="0.011993" G="-0.018308" B="0.691003" />
54     <cdm:RGB R="0.013995" G="-0.023760" B="0.903048" />
55     <cdm:RGB R="0.034050" G="-0.018254" B="0.692249" />

```

```

1      <cdm:RGB R="0.036746" G="-0.023689" B="0.904397" />
2      <cdm:RGB R="0.080545" G="-0.023830" B="0.908226" />
3      <cdm:RGB R="0.149752" G="-0.024452" B="0.915123" />
4      <cdm:RGB R="1.002593" G="0.985026" B="-0.073045" />
5      <cdm:RGB R="1.249149" G="-0.024436" B="-0.002194" />
6      <cdm:RGB R="1.252780" G="-0.025117" B="0.015342" />
7      <cdm:RGB R="1.257972" G="-0.026125" B="0.049916" />
8      <cdm:RGB R="1.271263" G="-0.029093" B="0.182298" />
9      <cdm:RGB R="1.279122" G="-0.031189" B="0.284927" />
10     <cdm:RGB R="1.002258" G="0.999362" B="0.998757" />
11     <cdm:RGB R="0.007818" G="-0.023805" B="0.902837" />
12     <cdm:RGB R="-0.001325" G="0.002612" B="-0.000199" />
13     <cdm:RGB R="0.004903" G="-0.018821" B="0.903364" />
14     <cdm:RGB R="-0.005430" G="-0.000728" B="0.906022" />
15     <cdm:RGB R="-0.024203" G="0.033431" B="0.912191" />
16     <cdm:RGB R="-0.042091" G="0.089818" B="-0.005932" />
17     <cdm:RGB R="-0.052335" G="0.086586" B="0.921578" />
18     <cdm:RGB R="-0.074175" G="0.159024" B="-0.010406" />
19     <cdm:RGB R="-0.090966" G="0.161809" B="0.932665" />
20     <cdm:RGB R="-0.117963" G="0.253649" B="-0.016500" />
21     <cdm:RGB R="-0.174730" G="0.376455" B="-0.024386" />
22     <cdm:RGB R="-0.204641" G="0.390245" B="0.954021" />
23     <cdm:RGB R="-0.331890" G="0.716723" B="-0.046182" />
24     <cdm:RGB R="-0.374737" G="0.740509" B="0.968358" />
25     <cdm:RGB R="-0.434478" G="0.938894" B="-0.060390" />
26     <cdm:RGB R="0.003567" G="-0.000129" B="-0.000031" />
27     <cdm:RGB R="0.013995" G="-0.023760" B="0.903048" />
28     <cdm:RGB R="0.011145" G="-0.018782" B="0.903675" />
29     <cdm:RGB R="0.001010" G="-0.000696" B="0.906548" />
30     <cdm:RGB R="0.017488" G="0.033476" B="0.912853" />
31     <cdm:RGB R="-0.045318" G="0.086662" B="0.922243" />
32     <cdm:RGB R="-0.083641" G="0.161924" B="0.933260" />
33     <cdm:RGB R="-0.133701" G="0.262246" B="0.944373" />
34     <cdm:RGB R="-0.196722" G="0.390440" B="0.954439" />
35     <cdm:RGB R="0.020512" G="-0.000542" B="-0.000142" />
36     <cdm:RGB R="0.034096" G="-0.018719" B="0.905266" />
37     <cdm:RGB R="0.024587" G="-0.000632" B="0.908671" />
38     <cdm:RGB R="0.006983" G="0.033602" B="0.915280" />
39     <cdm:RGB R="-0.019836" G="0.086907" B="0.924618" />
40     <cdm:RGB R="-0.057114" G="0.162314" B="0.935374" />
41     <cdm:RGB R="-0.106133" G="0.262787" B="0.946166" />
42     <cdm:RGB R="-0.168134" G="0.391122" B="0.955918" />
43     <cdm:RGB R="0.080545" G="-0.023830" B="0.908226" />
44     <cdm:RGB R="0.078174" G="-0.018848" B="0.909282" />
45     <cdm:RGB R="0.069557" G="-0.000692" B="0.913080" />
46     <cdm:RGB R="0.053304" G="0.033705" B="0.919825" />
47     <cdm:RGB R="0.028089" G="0.087255" B="0.928902" />
48     <cdm:RGB R="-0.007469" G="0.162946" B="0.939149" />
49     <cdm:RGB R="-0.054727" G="0.263702" B="0.949360" />
50     <cdm:RGB R="0.149752" G="-0.024452" B="0.915123" />
51     <cdm:RGB R="0.147668" G="-0.019423" B="0.916222" />
52     <cdm:RGB R="0.140009" G="-0.001093" B="0.920043" />
53     <cdm:RGB R="0.125292" G="0.033611" B="0.926547" />
54     <cdm:RGB R="0.102016" G="0.087566" B="0.935050" />
55     <cdm:RGB R="0.068642" G="0.163702" B="0.944526" />

```



```

1      <cdm:RGB R="0.023701" G="0.264902" B="0.953919" />
2      <cdm:RGB R="0.248626" G="-0.025756" B="0.924685" />
3      <cdm:RGB R="0.246801" G="-0.020644" B="0.925712" />
4      <cdm:RGB R="0.240032" G="-0.002034" B="0.929243" />
5      <cdm:RGB R="0.226828" G="0.033128" B="0.935165" />
6      <cdm:RGB R="0.205577" G="0.087658" B="0.942824" />
7      <cdm:RGB R="0.174609" G="0.164417" B="0.951306" />
8      <cdm:RGB R="0.132331" G="0.266234" B="0.959693" />
9      <cdm:RGB R="0.381162" G="-0.027864" B="0.936087" />
10     <cdm:RGB R="0.379554" G="-0.022644" B="0.936982" />
11     <cdm:RGB R="0.313886" G="0.164903" B="0.959216" />
12     <cdm:RGB R="0.274412" G="0.267520" B="0.966469" />
13     <cdm:RGB R="0.551062" G="-0.030854" B="0.948537" />
14     <cdm:RGB R="0.404402" G="0.399877" B="0.979322" />
15     <cdm:RGB R="0.761768" G="-0.034778" B="0.961424" />
16     <cdm:RGB R="1.016507" G="-0.039680" B="0.974324" />
17     <cdm:RGB R="1.247925" G="-0.024207" B="-0.006804" />
18     <cdm:RGB R="1.318331" G="-0.045598" B="0.986957" />
19     <cdm:RGB R="1.002593" G="0.985026" B="-0.073045" />
20     <cdm:RGB R="-0.017084" G="0.004071" B="0.105192" />
21     <cdm:RGB R="0.000276" G="-0.000778" B="0.024704" />
22     <cdm:RGB R="0.001189" G="-0.003497" B="0.124447" />
23     <cdm:RGB R="0.003835" G="-0.011408" B="0.425034" />
24     <cdm:RGB R="0.006525" G="-0.019382" B="0.730759" />
25     <cdm:RGB R="0.991838" G="0.854433" B="0.996305" />
26     <cdm:RGB R="-0.085296" G="0.980282" B="0.983420" />
27     <cdm:RGB R="0.997109" G="1.001108" B="1.001729" />
28     <cdm:RGB R="0.000038" G="-0.000093" B="0.002143" />
29     <cdm:RGB R="0.000087" G="-0.000230" B="0.006229" />
30     <cdm:RGB R="0.000627" G="-0.001818" B="0.062146" />
31     <cdm:RGB R="0.005522" G="-0.016416" B="0.616931" />
32     <cdm:RGB R="0.006525" G="-0.019382" B="0.730759" />
33     <cdm:RGB R="0.001838" G="-0.000115" B="0.001922" />
34     <cdm:RGB R="0.001968" G="-0.000231" B="0.005969" />
35     <cdm:RGB R="0.002147" G="-0.000441" B="0.013224" />
36     <cdm:RGB R="0.002370" G="-0.000761" B="0.024478" />
37     <cdm:RGB R="0.002634" G="-0.001207" B="0.040491" />
38     <cdm:RGB R="0.002944" G="-0.001796" B="0.062002" />
39     <cdm:RGB R="0.003302" G="-0.002545" B="0.089732" />
40     <cdm:RGB R="0.003715" G="-0.003473" B="0.124383" />
41     <cdm:RGB R="0.004188" G="-0.004596" B="0.166643" />
42     <cdm:RGB R="0.004728" G="-0.005932" B="0.217190" />
43     <cdm:RGB R="0.005341" G="-0.007497" B="0.276684" />
44     <cdm:RGB R="0.006813" G="-0.011384" B="0.425118" />
45     <cdm:RGB R="0.008657" G="-0.016392" B="0.617056" />
46     <cdm:RGB R="0.009736" G="-0.019358" B="0.730905" />
47     <cdm:RGB R="-0.080359" G="0.980412" B="0.983525" />
48     <cdm:RGB R="0.005604" G="-0.000209" B="0.001935" />
49     <cdm:RGB R="0.005786" G="-0.000307" B="0.006004" />
50     <cdm:RGB R="0.006041" G="-0.000495" B="0.013270" />
51     <cdm:RGB R="0.006363" G="-0.000795" B="0.024529" />
52     <cdm:RGB R="0.006745" G="-0.001225" B="0.040547" />
53     <cdm:RGB R="0.007184" G="-0.001801" B="0.062069" />
54     <cdm:RGB R="0.007679" G="-0.002542" B="0.089815" />
55     <cdm:RGB R="0.008231" G="-0.003463" B="0.124487" />

```

```

1      <cdm:RGB R="0.009523" G="-0.005912" B="0.217344" />
2      <cdm:RGB R="0.010273" G="-0.007474" B="0.276866" />
3      <cdm:RGB R="0.012011" G="-0.011356" B="0.425354" />
4      <cdm:RGB R="0.013011" G="-0.013710" B="0.515596" />
5      <cdm:RGB R="0.014108" G="-0.016361" B="0.617344" />
6      <cdm:RGB R="0.015308" G="-0.019327" B="0.731220" />
7      <cdm:RGB R="0.021793" G="-0.013690" B="0.516185" />
8      <cdm:RGB R="0.023073" G="-0.016337" B="0.617959" />
9      <cdm:RGB R="0.024454" G="-0.019298" B="0.731861" />
10     <cdm:RGB R="0.025943" G="-0.022591" B="0.858499" />
11     <cdm:RGB R="0.037804" G="-0.019298" B="0.732989" />
12     <cdm:RGB R="0.039522" G="-0.022583" B="0.859653" />
13     <cdm:RGB R="0.057988" G="-0.022628" B="0.861439" />
14     <cdm:RGB R="-0.012314" G="0.982157" B="0.984935" />
15     <cdm:RGB R="0.020932" G="0.982985" B="0.985609" />
16     <cdm:RGB R="0.110705" G="0.985131" B="0.987376" />
17     <cdm:RGB R="0.234831" G="0.987909" B="0.989706" />
18     <cdm:RGB R="0.311005" G="0.989512" B="0.991076" />
19     <cdm:RGB R="0.494043" G="0.993078" B="0.994197" />
20     <cdm:RGB R="0.721366" G="0.997001" B="0.997773" />
21     <cdm:RGB R="0.852925" G="0.999044" B="0.999710" />
22     <cdm:RGB R="0.997109" G="1.001108" B="1.001729" />
23     <cdm:RGB R="0.007645" G="-0.022679" B="0.857328" />
24     <cdm:RGB R="0.001742" G="-0.000069" B="-0.000016" />
25     <cdm:RGB R="0.010928" G="-0.022656" B="0.857495" />
26     <cdm:RGB R="0.005453" G="-0.000169" B="-0.000042" />
27     <cdm:RGB R="0.016619" G="-0.022623" B="0.857831" />
28     <cdm:RGB R="-0.071870" G="0.980634" B="0.983705" />
29     <cdm:RGB R="0.012251" G="-0.000330" B="-0.000086" />
30     <cdm:RGB R="-0.058141" G="0.980991" B="0.983991" />
31     <cdm:RGB R="-0.038488" G="0.981494" B="0.984397" />
32     <cdm:RGB R="0.038332" G="-0.000891" B="-0.000240" />
33     <cdm:RGB R="0.081991" G="-0.022759" B="0.863942" />
34     <cdm:RGB R="0.112193" G="-0.023008" B="0.867198" />
35     <cdm:RGB R="0.061770" G="0.983975" B="0.986421" />
36     <cdm:RGB R="0.168229" G="0.986444" B="0.988470" />
37     <cdm:RGB R="0.397243" G="0.991242" B="0.992576" />
38     <cdm:RGB R="0.601915" G="0.995006" B="0.995934" />
39     <cdm:RGB R="0.721366" G="0.997001" B="0.997773" />
40     <cdm:RGB R="0.849328" G="0.852650" B="0.994033" />
41     <cdm:RGB R="0.000000" G="0.000000" B="1.000000" />
42     </cdm:GamutBoundarySamples>
43     </cdm:MeasurementData>
44
45     </cdm:RGBVirtualDevice>
46
47 </cdm:ColorDeviceModel>

```

48

1

2 H. Standard Namespaces and Content Types

3 The following tables list the namespaces and content types used in XPS packages and XPS
4 Documents.

5 H.1 XML Namespace URIs

6 *Table H-1. Package-wide namespaces*

Description	Namespace URI
Content Types	http://schemas.openxmlformats.org/package/2006/content-types
Core Properties	http://schemas.openxmlformats.org/package/2006/metadata/core-properties
Digital Signatures	http://schemas.openxmlformats.org/package/2006/digital-signature
Relationships	http://schemas.openxmlformats.org/package/2006/relationships
Markup Compatibility	http://schemas.openxmlformats.org/markup-compatibility/2006

7 *Table H-2. XPS Document namespaces*

Description	Namespace URI
DiscardControl	http://schemas.microsoft.com/xps/2005/06/discard-control
Document Structure	http://schemas.microsoft.com/xps/2005/06/documentstructure
FixedDocument	http://schemas.microsoft.com/xps/2005/06
FixedDocumentSequence	http://schemas.microsoft.com/xps/2005/06
FixedPage	http://schemas.microsoft.com/xps/2005/06
Print Schema Framework	http://schemas.microsoft.com/windows/2003/08/printing/printschemaframework
Print Schema Keywords	http://schemas.microsoft.com/windows/2003/08/printing/printschemakeywords
Resource Dictionary (Key attribute)	http://schemas.microsoft.com/xps/2005/06/resource-dictionary-key
Signature Definitions	http://schemas.microsoft.com/xps/2005/06/signature-definitions
Story Fragments	http://schemas.microsoft.com/xps/2005/06/documentstructure

1 H.2 Content Types

2 The content types in the tables below MUST NOT include parameters. A consumer MUST treat
 3 the presence of parameters on these content types as an error when the affected part is
 4 accessed [M12.7].

5 *Table H-3. Package-wide content types*

Description	Content type
Core Properties part	application/vnd.openxmlformats-package.core-properties+xml
Digital Signature Certificate part	application/vnd.openxmlformats-package.digital-signature-certificate
Digital Signature Origin part	application/vnd.openxmlformats-package.digital-signature-origin
Digital Signature XML Signature part	application/vnd.openxmlformats-package.digital-signature-xmlsignature+xml
Relationships part	application/vnd.openxmlformats-package.relationships+xml

6 *Table H-4. XPS Document content types*

Description	Content type
FixedDocument	application/vnd.ms-package.xps-fixeddocument+xml
FixedDocumentSequence	application/vnd.ms-package.xps-fixeddocumentsequence+xml
FixedPage	application/vnd.ms-package.xps-fixedpage+xml
DiscardControl	application/vnd.ms-package.xps-discard-control+xml
DocumentStructure	application/vnd.ms-package.xps-documentstructure+xml
Font	application/vnd.ms-opentype
ICC profile	application/vnd.ms-color.iccprofile
JPEG image	image/jpeg
Obfuscated font	application/vnd.ms-package.obfuscated-opentype
PNG image	image/png
PrintTicket	application/vnd.ms-printing.printticket+xml
Remote resource dictionary	application/vnd.ms-package.xps-resourcedictionary+xml
StoryFragments	application/vnd.ms-package.xps-storyfragments+xml
TIFF image	image/tiff
Thumbnail part	image/jpeg or image/png
Windows Media Photo image	image/vnd.ms-photo

1 H.3 Relationship Types

2 *Table H-5. Package-wide relationship types*

Description	Relationship type
Core Properties	http://schemas.openxmlformats.org/package/2006/relationships/metadata/core-properties
Digital Signature	http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/signature
Digital Signature Certificate	http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/certificate
Digital Signature Origin	http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/origin
Thumbnail	http://schemas.openxmlformats.org/package/2006/relationships/metadata/thumbnail

3 *Table H-6. XPS Document relationship types*

Description	Relationship type
Digital Signature Definitions	http://schemas.microsoft.com/xps/2005/06/signature-definitions
DiscardControl	http://schemas.microsoft.com/xps/2005/06/discard-control
DocumentStructure	http://schemas.microsoft.com/xps/2005/06/documentstructure
PrintTicket	http://schemas.microsoft.com/xps/2005/06/printticket
Required Resource	http://schemas.microsoft.com/xps/2005/06/required-resource
Restricted Font	http://schemas.microsoft.com/xps/2005/06/restricted-font
StartPart	http://schemas.microsoft.com/xps/2005/06/fixerepresentation
StoryFragments	http://schemas.microsoft.com/xps/2005/06/storyfragments

1

2 I. Conformance Requirements

3 This annex is informative

4 This annex summarizes all conformance requirements for producers and consumers
5 implementing the XML Paper Specification. It is intended as a convenience; the text in the
6 referenced clause or subclause is considered normative in all cases.

7 In this annex, conformance requirements are divided into three tables per clause, respectively
8 containing the requirements that producers and consumers must follow, those that they should
9 follow, and those that are optional. Each conformance requirement is given a unique ID
10 comprised of a letter (M – MUST; S – SHOULD; O – OPTIONAL), a requirements group number,
11 and a unique ID within that clause. (§9 is the first clause that contains conformance
12 requirements, so that clause has a requirements group number of 1. Each subsequent clause
13 that contains conformance requirements has the next requirements group number, in
14 ascending order.) Producers and consumers can use these IDs to report error conditions. [If a
15 requirement is removed from this specification, its ID will not be reused for any newly added
16 requirement.](#)

17 Additionally, each table identifies who is burdened with enforcing or supporting the
18 requirement—the producer of content for that format or the consumer of content in that
19 format—marked with an “x” in the appropriate column. The consumer could simply be required
20 to validate that the producer correctly enforced a requirement in the generation of an XPS
21 Document; these cases are marked with a “v” instead of an “x”. In certain cases, a
22 requirement only applies to certain producers or consumers; these are marked with a
23 superscripted letter referenced at the end of the table.

24 Consumers must support the usage of conformance rules marked as “OPTIONAL” and
25 “SHOULD” only for producers if the consumer accesses the referenced feature. If a consumer or
26 producer does not access the referenced feature it must ignore the manifestation of the rule
27 without error.

28 In addition to the conformance requirements identified below, producers and consumers must
29 meet the conformance requirements described in the OPC.

30 Numerous entries in the Producer and Consumer columns of the tables in this annex have
31 superscripts, the meaning of which is, as follows:

32

Superscript	Meaning
A	Only applies to producers or consumers implementing digital signature features.
C	Only applies if Core Properties are used.
D	Only applies to producers that generate document structure or consumers that use it.
E	Only applies to consumers that are also producers

F	Only necessary if the referenced feature is used.
H	Only applies to producers that generate hyperlinks or consumers that implement hyperlinks.
K	Only applies to consumers that consume the XPS Document package head-first.
P	Only applies to printing consumers.
R	Only applies to producers or consumers that use DiscardControl parts.
S	Only applies to consumers that implement selection.
T	Only applies if thumbnails are used by the consumer.
U	Only required for a producer if the part is generated or for a consumer when accessing the part.

I.1 XPS Document Format

I.1.1 MUST Conformance Requirements

Table I-1. XPS Document format MUST conformance requirements

ID	Rule	Reference	Producer	Consumer
M1.1	XPS Documents MUST observe all conformance requirements of the OPC specification, except where specifically noted otherwise in this specification.	Clause 8	x	x
M1.2	The XPS Document format MUST use a ZIP archive for its physical model.	8.2	x	x

I.2 Parts and Relationships

I.2.1 MUST Conformance Requirements

Table I-2. Parts and Relationships MUST conformance requirements

ID	Rule	Reference	Producer	Consumer
M2.1	All content to be rendered MUST be contained in the XPS Document.	9.1	x	v
M2.2	Each part contained in an XPS Document MUST use only the appropriate content type.	9.1	x	v ^U

ID	Rule	Reference	Producer	Consumer
M2.3	An XPS Document MUST contain exactly one FixedDocumentSequence part per fixed payload.	9.1, 9.1.2	x	v
M2.4	An XPS Document MUST contain at least one FixedDocument part per fixed payload.	9.1	x	v
M2.5	An XPS Document MUST contain at least one FixedPage part per fixed payload.	9.1	x	v
M2.6	A <Glyphs> element in FixedPage markup MUST reference a Font part that exists in the XPS Document.	9.1	x	v
M2.7	An <ImageBrush> element in FixedPage markup MUST reference an Image part that exists in the XPS Document.	9.1	x	v
M2.8	If FixedPage markup references a Remote Resource Dictionary part, it MUST be included in the XPS Document	9.1	x	v
M2.9	This requirement was removed prior to Edition 1 of this specification.			
M2.10	Resources, which include fonts, images, color profiles, and remote resource dictionaries, that are referenced by URIs in FixedPage markup MUST use the Required Resource relationship from the FixedPage to the resource. If any resource references other resources, the indirectly required resource is also targeted by a Required Resource relationship from the FixedPage to the indirectly required resource.	9.1.1	x	v
M2.11	This requirement was removed prior to Edition 1 of this specification.			
M2.12	A Restricted Font relationship is REQUIRED for each print and preview font used, from the FixedDocument part to the preview and print Font part. When invoking editing functionality, a consumer that is also a producer MUST treat as an error any font with the print and preview licensing intent bit set for which no Restricted Font relationship has been added to the FixedDocument part. Printing and display-only consumers MUST consider an XPS Document valid, even if the producer failed to properly set the Restricted Font relationship.	9.1.1, 9.1.7.2, 9.1.7.4	x	v ^E
M2.13	Exactly one StartPart relationship is REQUIRED.	9.1.1	x	v
M2.14	The StartPart relationship MUST point from the package to the FixedDocumentSequence part that is the primary fixed payload root.	9.1, 9.1.2	x	v
M2.15	The order of <DocumentReference> elements in a FixedDocumentSequence part MUST be preserved by consumers that are also producers.	9.1.2		x ^E
M2.16	The order of <PageContent> elements in a FixedDocument MUST be preserved by consumers that are also producers.	9.1.3		x ^E
M2.17	JPEG image parts MUST contain images that conform to the	9.1.5.1	x	v ^U

ID	Rule	Reference	Producer	Consumer
	JPEG specification.			
M2.18	PNG image parts MUST contain images that conform to the PNG specification.	9.1.5.2	×	v ^U
M2.19	The PNG ancillary chunk tRNS MUST be supported.	9.1.5.2		×
M2.20	The PNG ancillary chunk iCCP MUST be supported.	9.1.5.2		×
M2.21	The PNG ancillary chunk sRGB MUST be ignored.	9.1.5.2		×
M2.22	The PNG ancillary chunk cHRM MUST be ignored.	9.1.5.2		×
M2.23	The PNG ancillary chunk gAMA MUST be ignored.	9.1.5.2		×
M2.24	The PNG ancillary chunk sBIT MUST be ignored.	9.1.5.2		×
M2.25	TIFF image parts MUST contain images that conform to the TIFF specification	9.1.5.3	×	v ^U
M2.26	XPS Document consumers MUST support baseline TIFF 6.0 with the tag values described in Table 9–5 for the specified TIFF image types, excepting the tags described in §9.1.5.3.	9.1.5.3		×
M2.27	If a TIFF file contains multiple image file directories (IFDs), consumers MUST use only the first IFD and ignore all others.	9.1.5.3		×
M2.28	XPS Document consumers MUST support TIFF images using CCITT bilevel encoding.	9.1.5.3		×
M2.29	XPS Document consumers MUST support CMYK TIFF images.	9.1.5.3		×
M2.30	XPS Document consumers MUST support TIFF images with associated alpha data. If the ExtraSamples tag is 1, the alpha is treated as pre-multiplied alpha. With an ExtraSamples tag of 2, the alpha is treated as non-pre-multiplied alpha.	9.1.5.3		×
M2.31	XPS Document consumers MUST support TIFF images using LZW compression.	9.1.5.3		×
M2.32	XPS Document consumers MUST support TIFF images using differencing predictors.	9.1.5.3		×
M2.33	XPS Document consumers MUST support TIFF images using JPEG compression (compression mode 6 only).	9.1.5.3		×
M2.34	XPS Document consumers MUST support TIFF images with an embedded ICC profile.	9.1.5.3		×
M2.35	Windows Media Photo image files MUST conform to the Windows Media Photo specification.	9.1.5.4	×	v ^U
M2.36	Each FixedPage part MUST NOT have more than one thumbnail part attached.	9.1.6	×	v ^T _⌘
M2.37	Thumbnails MUST be either JPEG or PNG images	9.1.6	×	v ^T
M2.38	If using a fragment in the FontURI attribute of the <Glyphs> element to indicate the font face to use from a TrueType Collection, the attribute value MUST be an integer between 0	9.1.7	×	v

ID	Rule	Reference	Producer	Consumer
	and $n-1$ inclusive, where n is the number of font faces in the TrueType Collection.			
M2.39	All fonts used in XPS Documents MUST adhere to the OpenType font format, which includes TrueType and CFF fonts. A subsetted font MUST still be a valid OpenType font file.	9.1.7.2, 9.1.7.1	×	✓
M2.40	Producers MUST honor the licensing rights specified in OpenType fonts by following the embedding and obfuscation mechanisms described in this specification.	9.1.7.2	×	
M2.41	Consumers MUST be able to process XPS Documents using any combination of the embedding and obfuscation mechanisms described in this specification (even if produced in violation of the production requirements).	9.1.7.2		×
M2.42	For fonts with "Restricted license embedding" licensing intent, producers MUST NOT embed the font.	9.1.7.2	×	
M2.43	For fonts with "Print and preview embedding" licensing intent, consumers MUST NOT edit or modify any part of the XPS Document markup or hierarchical structure from the FixedDocument containing such a font downwards.	9.1.7.2, 9.1.7.4		× ^E
M2.44	For fonts with "Print and preview embedding" licensing intent, producers MUST perform embedded font obfuscation.	9.1.7.2	×	
M2.45	For fonts with "Print and preview embedding" licensing intent, consumers MUST NOT extract or permanently install the font.	9.1.7.2		×
M2.46	For fonts with "Editable embedding" licensing intent, producers MUST perform embedded font obfuscation.	9.1.7.2	×	
M2.47	For fonts with "Editable embedding" licensing intent, consumers MUST NOT extract or permanently install the font.	9.1.7.2		×
M2.48	For fonts with "No subsetting" licensing intent, producers MUST perform embedded font obfuscation.	9.1.7.2	×	
M2.49	For fonts with "No subsetting" licensing intent, producers MUST NOT subset the font.	9.1.7.2	×	
M2.50	For fonts with "No subsetting" licensing intent, consumers MUST NOT extract or permanently install the font.	9.1.7.2		×
M2.51	For fonts with "Bitmap embedding only" licensing intent, producers MUST perform embedded font obfuscation for bitmap characters only. If no bitmap characters are present in the font, the producer MUST NOT embed the font.	9.1.7.2	×	
M2.52	For fonts with "Bitmap embedding only" licensing intent, consumers MUST NOT extract or permanently install the font.	9.1.7.2		×
M2.53	Producers and consumers MUST perform font obfuscation and de-obfuscation according to the steps described in §9.1.7.3.	9.1.7.3	×	×
M2.54	The last segment of the part name for an obfuscated font MUST	9.1.7.3	×	✓

ID	Rule	Reference	Producer	Consumer
	be the GUID generated during the font obfuscation process, with or without an extension.			
M2.55	When processing <Glyphs> elements, the consumer MUST select a cmap table from the OpenType font according to Table 1-8 in §9.1.7.5. All further processing for that font MUST use the selected cmap table.	9.1.7.5	x	x
M2.56	When processing <Glyphs> elements, if a WanSung, Big5, Prc, ShiftJis, or MacRoman cmap has been selected, the consumer MUST correctly map from Unicode codepoints in the UnicodeString attribute to the corresponding codepoints used by the cmap before looking up glyphs.	9.1.7.5		x
M2.57	When processing <Glyphs> elements that reference a cmap (3,0) encoding font, consumers MUST handle the case where the UnicodeString attribute contains character codes instead of PUA codepoints by computing the correct glyph index according to the general recommendations of the OpenType specification.	9.1.7.5		x
M2.58	Consumers MUST process all PrintTicket parts when an XPS Document is printed.	9.1.9.2, 9.1.9.3		x ^P
M2.59	A level-specific PrintTicket MUST contain only settings scoped to the current level and child levels. Job-level PrintTicket parts MUST contain only job-, document-, and page-scoped settings; document-level PrintTicket parts MUST contain only document-scoped and page-scoped settings; and page-level PrintTicket parts MUST contain only page-scoped settings. Print schema elements that interact between levels MUST be specified at the root of each level ticket. Each FixedDocumentSequence, FixedDocument, or FixedPage part MUST have no more than one attached PrintTicket.	9.1.9.2, 9.1.9.2.2, 9.1.9.3	x ^U	
M2.60	Consumers MUST process job-level, document-level and page-level settings of PrintTicket parts associated with FixedDocumentSequence parts.	9.1.9.2		x ^P
M2.61	Consumers MUST process document-level and page-level settings of PrintTicket parts associated with FixedDocument parts and MUST ignore job-level settings of PrintTicket parts associated with FixedDocument parts.	9.1.9.2		x ^P
M2.62	Consumers MUST process page-level settings of PrintTicket parts associated with FixedPage parts and MUST ignore job-level and document-level settings of PrintTicket parts associated with FixedPage parts.	9.1.9.2		x ^P
M2.63	When processing a PrintTicket, consumers MUST first remove all levels of PrintTicket content not applicable to the current element.	9.1.9.3		x ^P
M2.64	When processing a PrintTicket, consumers MUST second validate the PrintTicket according to the methods defined in the	9.1.9.3		x ^P

ID	Rule	Reference	Producer	Consumer
	PrintTicket Validation Checklist of the Print Schema documentation.			
M2.65	Following validation of a PrintTicket, the printing consumer MUST properly interpret the print settings according to the rules for merging two PrintTicket parts.	9.1.9.3		x ^P
M2.66	If there is no print setting merge conflict between different PrintTicket levels, a prefix-scoped element MUST be pushed down, or inherited, from a more general ticket to a more specific ticket. This case is isomorphic to the case where both tickets contain an identical element.	9.1.9.3		x ^P
M2.67	If there is a print setting merge conflict between different PrintTicket levels, the setting from the most specific ticket MUST take precedence.	9.1.9.3		x ^P
M2.68	Consumers MUST use semantic document structure provided in included DocumentStructure and StoryFragments parts in preference to any other analysis method of generating such structure.	9.1.11		x
M2.69	Consumers MUST support Markup Compatibility and Extensibility elements and attributes in DocumentStructure, FixedDocument, FixedDocumentSequence, FixedPage, Relationships, Remote Resource Dictionary, SignatureDefinitions, and StoryFragments parts. Before attempting to validate one of these parts against a schema, consumers MUST remove all Markup Compatibility and Extensibility elements and attributes, ignorable namespace declarations, and all ignored elements and attributes not defined in the expected version of XPS Document markup.	9.3.1, 9.3.2		x
M2.70	XML content MUST be encoded using either UTF-8 or UTF-16. If any such part includes an encoding declaration (as defined in §4.3.3 of the XML specification), that declaration MUST NOT name any encoding other than UTF-8 or UTF-16.	9.3.2	x	v
M2.71	DTD content MUST NOT be used in the XML markup defined in this specification, and consumers MUST treat the presence of DTD content as an error.	9.3.2	x	x
M2.72	XML content MUST be valid against the corresponding W3C XSD schema defined in this specification. In particular, the XML content MUST NOT contain elements or attributes drawn from namespaces that are not explicitly defined in the corresponding XSD unless the XSD allows elements or attributes drawn from any namespace to be present in particular locations in the XML markup.	9.3.2	x	v
M2.73	XML content MUST NOT contain elements or attributes drawn from "xml" or "xsi" namespaces unless they are explicitly defined in the W3C XSD schema or by other means in the	9.3.2	x	v

ID	Rule	Reference	Producer	Consumer
	specification.			
M2.74	Properties MUST NOT be set more than once, regardless of the syntax used to specify the value. In certain cases, they can be specified using either property attributes or property elements. Consumers MUST treat properties that are specified in both ways as an error.	9.3.3.2	x	v
M2.75	XPS Document markup MUST NOT use the xml:space attribute.	9.3.4	x	v
M2.76	The language of the contents of an XPS Document MUST be identified using the xml:lang attribute, the value of which is inherited by child and descendant elements. When the language of the contents is unknown and is required, the value "und" (undetermined) MUST be used.	9.3.5.1	x	
M2.77	Producers that generate a relationship MUST include the target part in the XPS Document for any of the following relationship types: DiscardControl, DocumentStructure, PrintTicket, Required Resource, Restricted Font, StartPart, StoryFragments, and Thumbnail. Consumers that access the target part of any relationship with one of these relationship types MUST generate an error if the part is not included in the XPS Document.	9.1.1	x ^u	v ^u
M2.78	Consumers MUST support JPEG images that contain the APP1 marker and interpret the EXIF color space correctly.	9.1.5.1 9.1.5.2		x
M2.79	XPS Document consumers MUST support TIFF images that include the EXIF IFD (tag 34665) as described in the EXIF specification. The EXIF color space MUST be interpreted correctly.	9.1.5.3		x
M2.80	Each <DocumentReference> element in a FixedDocumentSequence part MUST reference a FixedDocument part by relative URI.	9.1.2	x	
M2.81	Each <PageContent> element in a FixedDocument part MUST reference a FixedPage part by relative URI.	9.1.3	x	
M2.82	<ImageBrush> and <Glyphs> elements MUST reference Image and Font parts by relative URI.	9.1.4	x	
M2.83	If the ExtraSamples tag value is 0, the associated alpha data in this channel MUST be ignored	9.1.5.3		x

1 **I.2.2 SHOULD Conformance Requirements**2 *Table I-3. Parts and Relationships SHOULD conformance requirements*

ID	Rule	Reference	Producer	Consumer
S2.1	It is RECOMMENDED that there be exactly one Required Resource relationship from the FixedPage part for each resource referenced from the markup.	9.1.1	x	
S2.2	This requirement was removed prior to Edition 1 of this specification; its description is retained here for historical purposes. Each <DocumentReference> element in a FixedDocumentSequence part SHOULD reference a FixedDocument part by relative URI.	2.1.2	x	
S2.3	This requirement was removed prior to Edition 1 of this specification; its description is retained here for historical purposes. Each <PageContent> element in a FixedDocument part SHOULD reference a FixedPage part by relative URI.	2.1.3	x	
S2.4	This requirement was removed prior to Edition 1 of this specification; its description is retained here for historical purposes. <ImageBrush> and <Glyphs> elements SHOULD reference Image and Font parts by relative URI.	2.1.4	x	
S2.5	This requirement was removed prior to Edition 1 of this specification; its description is retained here for historical purposes. Color profiles embedded in image files SHOULD be used if present and compatible with this specification.	9.1.5		x
S2.6	It is RECOMMENDED that JPEG image part names end with the extension ".jpg".	9.1.5.1	x	
S2.7	The use of CMYK JPEG images is NOT RECOMMENDED. TIFF or Windows Media Photo images SHOULD be used instead to represent CMYK images.	9.1.5.1 15.3.4 15.3.7	x	
S2.8	It is RECOMMENDED that PNG image part names end with the extension ".png".	9.1.5.2	x	
S2.9	It is RECOMMENDED that TIFF image part names end with the extension ".tif".	9.1.5.3	x	
S2.10	Consumers SHOULD ignore unsupported TIFF tags (those not described in Table 9-5 and §9.1.5.3). Producers SHOULD NOT include unsupported tags.	9.1.5.3	x	x
S2.11	Given the wide variety of incompliant TIFF images in	9.1.5.3		x

ID	Rule	Reference	Producer	Consumer
	circulation, consumers SHOULD test as many different TIFF images as possible, correct common mistakes in TIFF images, and implement a reasonable recovery strategy when a problematic TIFF image is encountered.			
S2.12	It is RECOMMENDED that Windows Media Photo images end with the extension ".wdp".	9.1.5.4	x	
S2.13	It is RECOMMENDED that if thumbnails are used for pages, a thumbnail SHOULD be included for every page in the document.	9.1.6	x	
S2.14	Consumers SHOULD only process thumbnails associated via a package relationship from the package as a whole or via a relationship from a FixedPage part. Thumbnails attached to any other part SHOULD be ignored.	9.1.6		x
S2.15	Producers SHOULD use Unicode-encoded fonts.	9.1.7, 9.1.7.5	x	
S2.16	For fonts with "Installable embedding" licensing intent, producers SHOULD perform embedded font obfuscation.	9.1.7.2	x	
S2.17	For fonts with "Installable embedding" licensing intent, consumers SHOULD NOT extract or permanently install the font.	9.1.7.2		x
S2.18	For fonts with "Restricted license embedding" licensing intent, producers SHOULD generate a path filled with an image brush referencing an image of rendered characters and SHOULD include the actual text in the AutomationProperties.Name attribute of the <Path> element.	9.1.7.2	x	
S2.19	Although the licensing intent allows embedding of non-obfuscated fonts and installation of the font on a remote client system under certain conditions, this is NOT RECOMMENDED in XPS Documents. Instead, producers SHOULD always perform font obfuscation, and consumers SHOULD never extract or permanently install fonts.	9.1.7.3	x	x
S2.20	It is RECOMMENDED that the extension of an obfuscated Font part name be ".odtff" for TrueType fonts and ".odttc" for TrueType collections.	9.1.7.3	x	
S2.21	Producers SHOULD include only PrintTicket settings that support portability of the XPS Document.	9.1.9.1	x	
S2.22	Producers SHOULD only attach PrintTicket parts containing only document-level and page-level settings with FixedDocument parts.	9.1.9.2	x	
S2.23	Producers SHOULD only attach PrintTicket parts containing only page-level settings with FixedPage parts.	9.1.9.2 9.1.9.3	x	
S2.24	The FixedDocumentSequence part SHOULD follow the part name recommendation "</FixedDocSeq>.fdseq" where	9.2	x	

ID	Rule	Reference	Producer	Consumer
	<i><FixedDocSeq></i> is the name of the FixedDocumentSequence. The FixedDocumentSequence SHOULD use the extension ".fdseq".			
S2.25	A FixedDocument part SHOULD follow the part name recommendation "/Documents/<n>/<FixedDocument>.fdoc" where <n> is a numeral that indicates the ordinal position of the fixed document in the fixed document sequence and <FixedDocument> is the name of the fixed document. FixedDocument parts SHOULD use the extension ".fdoc".	9.2	x	
S2.26	A FixedPage part SHOULD follow the part name recommendation "/Documents/<n>/Pages/<m>.fpage" where <n> represents the document that includes this page and <m> is the page number. FixedPage parts SHOULD use the extension ".fpage".	9.2	x	
S2.27	<p>A resource that is specific to a particular document SHOULD follow the part name recommendation "/Documents/<n>/Resources/<Resource>" where <n> is the document that uses the resource and <Resource> is the segments identifying the particular resource.</p> <p>A resource that is shared across multiple documents SHOULD follow the part name recommendation "/Resources/<Resource>".</p> <p>A Font resource SHOULD use "Fonts/<FontName>.<FontExt>" for its <Resource> value, where <FontExt> SHOULD be either ".ttf" or ".odttf" for non-obfuscated and obfuscated fonts respectively.</p> <p>An Image resource SHOULD use "Images/<ImageName>.<ImageExt>" for its <Resource> value, where <ImageExt> is the correct extension for the image type.</p> <p>A Remote Resource Dictionary resource SHOULD use "Dictionaries/<DictName>.dict" for its <Resource> value. A Remote Resource Dictionary SHOULD use ".dict" as its extension.</p> <p><FontName>, <ImageName>, and <DictName> SHOULD be a string representation of a GUID value if the resource is a shared resource.</p>	9.2	x	
S2.28	A DocumentStructure part SHOULD follow the part name recommendation "/Documents/<n>/Structure/<DocStruct>.struct" where <n> is the fixed document that the document structure applies to. DocumentStructure parts SHOULD use the extension ".struct".	9.2	x	
S2.29	A StoryFragments part SHOULD follow the part name recommendation	9.2	x	

ID	Rule	Reference	Producer	Consumer
	<p>"/Documents/<n>/Structure/Fragments/<m>.frag" where <n> is the fixed document that contains the story fragments and <m> is the page number the StoryFragments part applies to. StoryFragments parts SHOULD use the extension ".frag".</p>			
S2.30	<p>ICC profile parts SHOULD follow the part name recommendation "/Documents/<n>/Metadata/<ProfileName>.<ProfileExt>" where <n> is the fixed document that contains the profile. ICC profile parts shared across multiple documents SHOULD follow the part name recommendation "/Metadata/<ProfileName>.<ProfileExt>". In this case, <ProfileName> SHOULD be a string representation of a GUID value. The <ProfileExt> SHOULD be appropriate to the color profile type, such as ".icm".</p>	9.2	x	
S2.31	<p>Thumbnail parts SHOULD follow the part name recommendation "/Documents/<n>/Metadata/<ThumbName>.<ThumbExt>" where <n> is the fixed document that contains the profile. If the Thumbnail part represents the package as a whole, it SHOULD follow the part name recommendation "/Metadata/<ThumbName>.<ThumbExt>". In this case, <ThumbName> SHOULD be a string representation of a GUID value. The <ThumbExt> SHOULD be appropriate to the image type, either ".png" or ".jpg".</p>	9.2	x	
S2.32	<p>PrintTicket part names associated with the entire job SHOULD be associated via relationship with the FixedDocumentSequence part and follow the part name recommendation "/Metadata/<PrintTicketName>.xml". PrintTicket parts associated with a particular fixed document or fixed page SHOULD follow the part name recommendation "/Documents/<n>/Metadata/<PrintTicketName>.xml", where <n> is the fixed document that uses these parts. PrintTicket parts SHOULD use the extension ".xml".</p>	9.2	x	
S2.33	<p>The names of any non-standard parts that are associated with a particular fixed document SHOULD follow the part name recommendation "/Documents/<n>/Other/<PartName>", where <n> is the fixed document to which the part belongs.</p>	9.2	x	
S2.34	<p>Consumers SHOULD support JPEG images that contain JFIF-specified APP0 and ICC-specified APP2 markers.</p>	9.1.5.1	x	

ID	Rule	Reference	Producer	Consumer
S2.35	If the referenced font part is a TrueType Collection, then if the fragment portion of the URI is not recognised as a valid integer, consumers SHOULD generate an error.	9.1.7		x

1 I.2.3 OPTIONAL Conformance Requirements

2 Table I-4. Parts and Relationships OPTIONAL conformance requirements

ID	Rule	Reference	Producer	Consumer
O2.1	Thumbnail parts MAY be included in an XPS Document	9.1	x	x
O2.2	PrintTicket parts MAY be included in an XPS Document.	9.1	x	x
O2.3	ICC Profile parts MAY be included in an XPS Document.	9.1	x	x
O2.4	DocumentStructure parts MAY be included in an XPS Document.	9.1	x	x
O2.5	StoryFragments parts MAY be included in an XPS Document.	9.1	x	x
O2.6	SignatureDefinitions parts MAY be included in an XPS Document.	9.1	x	x
O2.7	DiscardControl parts MAY be included in an XPS Document.	9.1	x	x
O2.8	A Core Properties relationship MAY be included in an XPS Document, from the package to the Core Properties part.	9.1.1	x	x
O2.9	A Digital Signatures Origin relationship MAY be included in an XPS Document, from the package to the Digital Signature Origin part.	9.1.1	x	x
O2.10	Digital Signature relationships MAY be included in an XPS Document, from the Digital Signature Origin part to a Digital Signature XML Signature part.	9.1.1	x	x
O2.11	Digital Signature Certificate relationships MAY be included in an XPS Document, from a Digital Signature XML Signature part to the Digital Signature Certificate part.	9.1.1	x	x
O2.12	Digital Signature Definitions parts MAY be included in an XPS Document, from a FixedDocument part to the Digital Signature Definitions part.	9.1.1	x	x
O2.13	DiscardControl relationships MAY be included in an XPS Document, from the package to a DiscardControl part.	9.1.1	x	x
O2.14	DocumentStructure relationships MAY be included in an XPS Document, from a FixedDocument part to the DocumentStructure part.	9.1.1	x	x
O2.15	PrintTicket relationships MAY be included in an XPS Document, from a FixedDocumentSequence, FixedDocument, or FixedPage part to a PrintTicket part.	9.1.1	x	x
O2.16	StoryFragments relationships MAY be included in an XPS Document, from a FixedPage part to a StoryFragments part.	9.1.1	x	x

O2.17	Thumbnail relationships MAY be included in an XPS Document, from the package to an Image part or from a FixedPage part to an Image part.	9.1.1	×
O2.18	Color Profiles MAY be embedded in image files.	9.1.5	×
O2.19	Thumbnail images MAY be attached to a FixedPage part using a Thumbnail relationship.	9.1.6	×
O2.20	Fonts MAY be subsetted based on glyph usage.	9.1.7.1	×
O2.21	Producers MAY use a 128-bit random number instead of a true GUID for an obfuscated font name.	9.1.7.3	×
O2.22	An obfuscated Font part MAY have an arbitrary extension.	9.1.7.3	×
O2.23	Producers MAY add digital signature requests and instructions to an XPS Document in the form of signature definitions.	9.1.10	×
O2.24	A producer MAY sign against an existing signature definition to provide additional signature information.	9.1.10	×
O2.25	A recipient of an XPS Document MAY also sign it against a signature definition.	9.1.10	×
O2.26	This requirement was removed prior to Edition 1 of this specification.		
O2.27	Consumers MAY provide an algorithmic construction of the structure of an XPS Document based on a page-layout analysis, provided such structure is not explicitly provided in DocumentStructure and StoryFragments parts.	9.1.11	×
O2.28	A resource that is intended to be used across multiple fixed documents MAY be named according to the guidelines for shared resources.	9.2	×
O2.29	Producers MAY include Markup Compatibility and Extensibility elements and attributes in DocumentStructure, FixedDocument, FixedDocumentSequence, FixedPage, Relationships, Remote Resource Dictionary, SignatureDefinitions, and StoryFragments parts.	9.3.1	×
O2.30	Wherever a single whitespace character is allowed in XPS Document markup, multiple whitespace characters MAY be used (unless explicitly restricted by a pattern restriction in the corresponding schema).	9.3.4	×
O2.31	Attributes in XPS Document markup that specify comma-delimited attribute values MAY, unless specified otherwise, OPTIONALLY include whitespace characters preceding or following the comma.	9.3.4	×
O2.32	Where the XPS Document schema specifies attributes of types that allow whitespace collapsing, leading and trailing whitespace in the attribute value MAY be used along with other whitespace that relies on the whitespace collapsing behavior specified in the XML Schema Specification.	9.3.4	×

1 **I.3 Documents**2 **I.3.1 MUST Conformance Requirements**3 *Table I-5. Document MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M3.1	The order of <DocumentReference> elements MUST match the order of the documents in the fixed document sequence.	10.1	x	
M3.2	The Source attribute of the <DocumentReference> element MUST specify a FixedDocument part within the XPS Document.	10.1.1	x	v
M3.3	Producers MUST NOT produce a document with multiple <DocumentReference> elements that reference the same fixed document.	10.1.1	x	v
M3.4	The order of <PageContent> elements MUST match the order of the pages in the document.	10.2	x	
M3.5	The Source attribute of the <PageContent> element MUST specify a FixedPage part within the XPS Document.	10.2.1	x	v
M3.6	Producers MUST NOT produce markup where a <PageContent> element references the same fixed page referenced by any other <PageContent> element in the entire XPS Document, even in other fixed documents within the fixed payload.	10.2.1	x	v
M3.7	If the attribute is specified, the BleedBox BleedOriginX value MUST be less than or equal to 0.	10.3.1	x	v ^F
M3.8	If the attribute is specified, the BleedBox BleedOriginY value MUST be less than or equal to 0.	10.3.1	x	v ^F
M3.9	If the attribute is specified, the BleedBox BleedWidth value MUST be greater than or equal to the Width attribute value plus the absolute value of the BleedBox BleedOriginX value.	10.3.1	x	v ^F
M3.10	If the attribute is specified, the BleedBox BleedHeight value MUST be greater than or equal to the fixed page Height value plus the absolute value of the BleedBox BleedOriginY value.	10.3.1	x	v ^F
M3.11	If the attribute is specified, the ContentBox ContentOriginX value MUST be greater than or equal to 0 and less than the fixed page Width attribute value	10.3.2	x	v ^F
M3.12	If the attribute is specified, the ContentBox ContentOriginY value MUST be greater than or equal to 0 and less than the fixed page Height attribute value.	10.3.2	x	v ^F
M3.13	If the attribute is specified, the ContentBox ContentWidth value MUST be less than or equal to the difference between the fixed page Width attribute value and the ContentBox ContentOriginX value.	10.3.2	x	v ^F
M3.14	If the attribute is specified, the ContentBox ContentHeight value MUST be less than or equal to the difference between the fixed page Height attribute value and the ContentBox ContentOriginY value.	10.3.2	x	v ^F

M3.15	When rendering a fixed page for printing, consumers MUST be aware of the interaction between the fixed page markup and the PrintTicket settings.	10.3.4	x	v ^P
M3.16	When the PrintTicket specifies the page scaling option FitApplicationBleedSizeToPageImageableSize, printing consumers MUST scale the bleed box (producer bleed size) to the PageImageableSize, preserving the aspect ratio.	10.3.4.1		x ^P
M3.17	When the PrintTicket specifies the page scaling option FitApplicationContentSizeToPageImageableSize, printing consumers MUST scale the content box (producer content size) to the PageImageableSize, preserving the aspect ratio.	10.3.4.2		x ^P
M3.18	When the PrintTicket specifies the page scaling option FitApplicationMediaSizeToPageImageableSize, printing consumers MUST scale the height and width (producer media size) to the PageImageableSize, preserving the aspect ratio.	10.3.4.3		x ^P
M3.19	When the PrintTicket specifies the page scaling option FitApplicationMediaSizeToPageMediaSize, printing consumers MUST scale the height and width (producer media size) to the PageMediaSize, preserving the aspect ratio.	10.3.4.4		x ^P
M3.20	The x:Key attribute of the <Canvas> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	10.4	x	v

1 **I.3.2 SHOULD Conformance Requirements**

2 *Table I-6. Document SHOULD conformance requirements*

ID	Rule	Reference	Producer	Consumer
S3.1	Specifying a ContentBox attribute for the <FixedPage> element is RECOMMENDED.	10.3	x	
S3.2	Invalid bleed box specifications SHOULD be ignored in favor of the default bleed box.	10.3.1		x
S3.3	Invalid content box specifications SHOULD NOT be rendered and SHOULD generate an error.	10.3.1		x
S3.4	In the absence of media scaling, the fixed page content is imaged directly to the physical media with the origin of the fixed page aligned with the origin of the physical media size. Any fixed page content that extends beyond the dimension of the physical media size SHOULD be clipped.	10.3.4		x ^P

1 I.4 Graphics

2 I.4.1 MUST Conformance Requirements

3 *Table I-7. Graphics MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M4.1	The x:Key attribute of the <Path> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	11.1	x	v
M4.2	The x:Key attribute of the <PathGeometry> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	11.2.1.1	x	v
M4.3	A <PathGeometry> element contains a set of path figures specified either with the Figures attribute or with a child <PathFigure> element. Producers MUST NOT specify the path figures of a geometry with both the Figures attribute and a child <PathFigure> element.	11.2.1.1	x	v

4 I.4.2 SHOULD Conformance Requirements

5 *Table I-8. Graphics SHOULD conformance requirements*

ID	Rule	Reference	Producer	Consumer
S4.1	Line segments and curve segments SHOULD NOT be specified as zero-length.	11.2.2.1	x	

6 I.4.3 OPTIONAL Conformance Requirements

7 *Table I-9. Graphics OPTIONAL conformance requirements*

ID	Rule	Reference	Producer	Consumer
O4.1	Consumers or viewers that perform anti-aliasing MAY "snap" those control points of the path that are situated on the path bounding box to whole device pixels if the ignorable SnapsToDevicePixels attribute is specified as true.	4.1		x

1 **I.5 Text**2 **I.5.1 MUST Conformance Requirements**3 *Table I-10. Text MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M5.1	If the CaretStops attribute is missing from the <Glyphs> element, a consumer MUST interpret the text as having a caret stop between each Unicode UTF-16 code unit and at the beginning and end of the text.	12.1		x ^s
M5.2	If the UnicodeString attribute of the <Glyphs> element specifies an empty string ("{}") and the Indices attribute is not specified or is empty, the consumer MUST generate an error.	12.1	x	v
M5.3	The x:Key attribute of the <Glyphs> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	12.1	x	v
M5.4	The sum of the code unit counts for all the GlyphMapping entries in the Indices attribute MUST NOT exceed the number of UTF-16 code units in the UnicodeString attribute if the UnicodeString attribute is specified and does not contain an empty value ("{}"). If a ClusterMapping is not specified within a GlyphMapping entry, the code unit count is 1. If the Indices attribute specifies a GlyphIndex that does not exist in the font, the consumer MUST generate an error.	12.1.3	x	v
M5.5	If there is not a one-to-one mapping between code units in the UnicodeString attribute and the glyph indices, the GlyphIndex value in the Indices attribute MUST be specified.	12.1.3	x	
M5.6	The AdvanceWidth of the Indices attribute MUST be calculated as the exact unrounded origin of the subsequent glyph minus the sum of the calculated (that is, rounded) advance widths of the preceding glyphs.	12.1.3	x	
M5.7	A UnicodeString attribute value that begins with an open brace ("{"}) MUST be escaped with a prefix of "{". If a UnicodeString attribute value starts with "{", consumers MUST ignore those first two characters in processing the UnicodeString and in calculating index positions for the characters of the UnicodeString.	12.1.4	x	x
M5.8	This requirement was removed prior to Edition 1 of this specification.			
M5.9	If the UnicodeString attribute contains a Unicode code unit that cannot be mapped to a glyph index via a cmap table in the font and there is no corresponding GlyphIndex entry in the Indices attribute, the consumer MUST display the .notdef glyph	12.1.4	✘	x
M5.10	In the absence of entries in the Indices attribute to override the Unicode code units in the UnicodeString attribute value, consumers MUST treat Unicode control marks in the UnicodeString attribute like ordinary characters and render the glyphs to which the Unicode	12.1.4		x

	control marks are mapped in the CMAP table.			
M5.11	Because advance-widths, glyph indices, and caret-stops are associated with the generated Unicode string, consumers MUST NOT normalize the UnicodeString attribute value to produce an internal representation.	12.1.4	x	
M5.12	Producers MUST lay out algorithmically emboldened glyphs using advance widths that are 2% of the em size larger than when not algorithmically emboldened.	12.1.5	x	
M5.13	Consumers MUST implement the effect of algorithmic emboldening such that the black box of the glyph grows by 2% of the em size. When advance widths are omitted from the markup and the glyphs are algorithmically emboldened, the advance widths obtained from the horizontal metrics font table (if IsSideways is false) or the vertical metrics font table (if IsSideways is true) of the font MUST be increased by 2% of the em size.	12.1.5	x	
M5.14	Producers MUST lay out algorithmically italicized glyphs using exactly the same advance widths as when not algorithmically italicized.	12.1.5	x	
M5.15	Producers MUST NOT specify text that is both right-to-left (BidiLevel attribute value of 1) and vertical (IsSideways attribute set to true).	12.1.6.2	x	v
M5.16	If a consumer does not understand the specified device font name, it MUST render the embedded version of the font.	12.1.7		x ^P
M5.17	When rendering a printer device font, consumers MUST use the UnicodeString attribute and ignore the glyph index components of the Indices attribute.	12.1.7		x ^{FP}
M5.18	When rendering a printer device font, consumers MUST still honor the advance width and x,y offset values present in the Indices attribute.	12.1.7		x ^P
M5.19	For producers, a <Glyphs> element with a specified device font name MUST have exactly one Indices glyph per character in the UnicodeString attribute. Its Indices attribute MUST NOT include any cluster specifications. If the Indices attribute includes a cluster mapping, the consumer MUST NOT use the device font name and MUST render the embedded version of the font.	12.1.7	x	v ^{FP}
M5.20	For producers of a <Glyphs> element with a specified device font name, each of the Indices glyphs MUST include a specified advance width and MUST include specified x and y offset values if they are non-zero.	12.1.7	x	
M5.21	This requirement was removed prior to Edition 1 of this specification.			
M5.22	If there are insufficient flags in the CaretStops attribute value to correspond to all the UTF-16 code units in the UnicodeString attribute value, all remaining UTF-16 code units in the Unicode string MUST be considered valid caret stops.	12.1.9		x ^S
M5.23	If the Indices attribute is specified, the values provided MUST be used in preference to values determined from the UnicodeString attribute alone.	12.1.3	x	
M5.24	If the Indices attribute specifies a GlyphIndex that does not exist in	12.1.3	x	v

[the font, the consumer MUST generate an error.](#)

1 **I.5.2 SHOULD Conformance Requirements**

2 *Table I-11. Text SHOULD conformance requirements*

ID	Rule	Reference	Producer	Consumer
S5.1	The value of the CaretStops attribute SHOULD indicate that the caret cannot stop in front of most combining marks and the second UTF-16 code unit of UTF-16 surrogate pairs.	12.1	x	
S5.2	If producers include control marks in the Unicode string, they SHOULD include an Indices attribute to specify glyph indices and/or character-to-glyph mapping information for the control marks.	12.1.4	x	
S5.3	If alternate vertical character representations are available in the font, the producer SHOULD use those in preference to the IsSideways attribute and provide their glyph indices in the Indices attribute.	12.1.6	x	
S5.4	Producers SHOULD NOT produce markup that will result in different rendering between consumers using the embedded font to render and consumers using the device font to render.	12.1.7	x	
S5.5	Specifying a UnicodeString for <Glyphs> elements is RECOMMENDED, as it supports searching, selection, and accessibility.	12.1.4	x	

3 **I.5.3 OPTIONAL Conformance Requirements**

4 *Table I-12. Text OPTIONAL conformance requirements*

ID	Rule	Reference	Producer	Consumer
O5.1	Producers MAY include Unicode control marks in the Unicode string. Such marks include control codes, layout controls, invisible operators, deprecated format characters, variation selectors, non-characters, and specials, according to their definition within the Unicode specification.	12.1.4	x	
O5.2	Producers MAY choose to generate UnicodeString attribute values that are not normalized by any Unicode-defined algorithm.	12.1.4	x	
O5.3	Consumers that understand the device font name MAY ignore the embedded font and use the device-resident version.	12.1.7		x ^P
O5.4	Glyph indices MAY be omitted from markup where there is a one-to-one mapping between the positions of Unicode scalar values in the UnicodeString attribute and the positions of glyphs in the glyph string and the glyph index is the value in selected character mapping table of the font.	12.1.10.1	x	
O5.5	Glyph advance widths MAY be omitted from markup where the advance width desired is specified in the font tables, once adjusted for	12.1.10.2	x	

algorithmic emboldening.

O5.6	Glyph horizontal and vertical offsets MAY be omitted from markup where the offset is 0.0.	12.1.10.2	x	
------	---	-----------	---	--

1 I.6 Brushes

2 I.6.1 MUST Conformance Requirements

3 *Table I-13. Brushes MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M6.1	The x:Key attribute of the <SolidColorBrush> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	13.1	x	v
M6.2	The x:Key attribute of the <ImageBrush> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	13.2	x	v
M6.3	An <ImageBrush> element MUST reference a JPEG, PNG, TIFF-6.0, or Windows Media Photo Image part within the XPS Document package.	13.2	x	v
M6.4	The x:Key attribute of the <VisualBrush> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	13.3	x	v
M6.5	The x:Key attribute of the <LinearGradientBrush> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	13.5	x	v
M6.6	The x:Key attribute of the <RadialGradientBrush> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	13.6	x	v

4 I.7 Common Properties

5 I.7.1 MUST Conformance Requirements

6 *Table I-14. Common properties MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M7.1	Individual resource values MUST be specified within a resource dictionary.	14.2	x	
M7.2	Namespace prefixes in resource definitions MUST apply in the context of the definition, rather than in the context of the resource reference.	14.2.3	x	x

M7.3	An xml:lang attribute within a resource definition MUST be interpreted in the context of the resource reference, not the resource definition.	14.2.3	x	x
M7.4	A remote resource dictionary MUST follow the requirements that apply to inline resource dictionaries.	14.2.3.1	x	v
M7.5	A remote resource dictionary MUST NOT contain any resource definition children that reference another remote resource dictionary.	14.2.3.1	x	v
M7.6	A <ResourceDictionary> element that specifies a remote resource dictionary in its Source attribute MUST NOT contain any resource definition children.	14.2.3.1	x	v
M7.7	Inline references to fonts or images in remote resource dictionary entries MUST be interpreted with the same base URI as the Remote Resource Dictionary part, not from the base URI of the part referring to the particular remote resource dictionary entry.	14.2.3.1	x	x
M7.8	When a resource definition references a previously defined resource with the same name in an ancestor resource dictionary, the reference MUST be resolved before the redefined resource is added to the dictionary	14.2.5	x	x
M7.9	If a resource definition references another resource, the reference MUST be resolved in the context of the resource definition, not in the context of the resource use.	14.2.5	x	x
M7.10	If a resource dictionary contains Markup Compatibility and Extensibility elements and attributes, the processing of the Markup Compatibility and Extensibility markup MUST occur in the context of the definition of the resource dictionary, not in the context of resource references.	14.2.6	x	x
M7.11	The x:Key attribute of the <MatrixTransform> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a resource dictionary.	14.4.1	x	v

1 **I.7.2 OPTIONAL Conformance Requirements**

2 *Table I-15. Common properties OPTIONAL conformance requirements*

ID	Rule	Reference	Producer	Consumer
O7.1	Resource dictionaries MAY be specified in separate parts (called remote resource dictionaries) and referenced from within the <FixedPage.Resources> or <Canvas.Resources> property element.	14.2	x	
O7.2	A resource definition MAY reference another resource defined prior to the point of reference, including a resource previously within the same resource dictionary.	14.2.3	x	
O7.3	If the resource dictionary does not appear in a separate part, a resource definition MAY reference a previously defined resource in a resource dictionary of a parent or ancestor <Canvas> or <FixedPage> element.	14.2.3	x	

[O7.4 This requirement was removed prior to Edition 1 of this specification.](#)

07.5	The resource dictionary of a <Canvas> element MAY re-use (and thus override within the scope of the re-use) an x:Key value defined in the resource dictionary of a parent or ancestor <Canvas> or <FixedPage> element.	14.2.5	x
07.6	A resource definition MAY reference a previously defined resource with the same name that is defined in an ancestor resource dictionary.	14.2.5	x

1 I.8 Color

2 I.8.1 MUST Conformance Requirements

3 Table I-16. Color MUST conformance requirements

ID	Rule	Reference	Producer	Consumer
M8.1	Consumers MUST support sRGB colors (8 bit-per-channel) in vector data, with and without alpha.	15.1		x
M8.2	Consumers MUST support sRGB colors in image data, using the JPEG, PNG, TIFF, or Windows Media Photo image formats.	15.1		x
M8.3	Consumers MUST support scRGB color specification in vector data, with and without alpha.	15.1		x
M8.4	Consumers MUST support scRGB colors in image data, using the Windows Media Photo image format.	15.1		x
M8.5	Consumers MUST support CMYK colors in vector data.	15.1		x
M8.6	Consumers MUST support CMYK colors in image data, using the TIFF or Windows Media Photo image formats.	15.1		x
M8.7	Consumers MUST support N-Channel colors in vector data.	15.1		x
M8.8	Consumers MUST support N-Channel colors in image data, using the Windows Media Photo image format.	15.1		x
M8.9	Consumers MUST support profiles compliant with ICC.1:2001-04 ICC Version 2 profiles for 3-, 4-, 5-, 6-, 7-, and 8-channel color.	15.1		x
M8.10	Consumers MUST support profiles compliant with ICC.1:2001-04 ICC Version 2 profiles with a Windows Color System (WCS) profile embedded as a private tag, either by handling or ignoring the WCS profile.	15.1		x
M8.11	Consumers MUST inspect the PageDeviceColorSpaceProfileURI PrintTicket setting to determine that this particular color specification is a native device color and MUST NOT be color-managed according to the included profile unless forced to do so for transparency effects.	15.1.7		x ^{FP}
M8.12	XPS producers and consumers MUST provide color management using ICC profiles conforming to the requirements of the ICC Color Profile specification, ICC.1:2001-04, for color spaces other than sRGB and scRGB.	15.1.8	x ^U	v ^U
M8.13	All ICC profiles used in XPS Documents MUST either be an Input	15.1.8	x ^U	v

	profile, an Output profile, a Monitor (RGB) profile, or a ColorSpace Conversion profile.			
M8.14	Real numbers specified for color channel values of scRGB and ContextColor colors MUST NOT use exponent forms of numbers.	15.2	x	v
M8.15	Although alpha values smaller than 0.0 and larger than 1.0 can be specified in scRGB images, the alpha values MUST be clamped to the valid range from 0.0 to 1.0 before any further processing.	15.2.2		x
M8.16	Although alpha values smaller than 0.0 and larger than 1.0 can be specified in CMYK images, the alpha values MUST be clamped to the valid range from 0.0 to 1.0 before any further processing.	15.2.3		x
M8.17	For N-Channel colors, the context color MUST specify a number of channel float values equal to the number of channels in the profile, setting unused ones to 0.0.	15.2.4	x	v
M8.18	Although alpha values smaller than 0.0 and larger than 1.0 can be specified in N-Channel images, the alpha values MUST be clamped to the valid range from 0.0 to 1.0 before any further processing.	15.2.4		x
M8.19	The profile for a named color MUST have 3, 4, 5, 6, 7 or 8 channels (and an <i>n</i> CLR signature, where <i>n</i> is the number of channels), mapping to a valid PCS.	15.2.5	x ^U	v
M8.20	Although alpha values smaller than 0.0 and larger than 1.0 can be specified in named color images, the alpha values MUST be clamped to the valid range from 0.0 to 1.0 before any further processing.	15.2.5		x
M8.21	Consumers that do not understand a named color MUST compute a color approximation through ICM-compliant color management functions using the specified profile.	15.2.5		x
M8.22	The color name specified by the DocumentImpositionColor PrintTicket setting MUST be matched only to profiles containing exactly one non-zero-length colorant name in the profile's colorantTable.	15.4		x ^{FP}
M8.23	The color name specified by the DocumentImpositionColor setting serves as a label for that color only and MUST NOT be matched against any Named Colors known by the consumer.	15.4		x ^{FP}
M8.24	The comparison of the color name specified by the DocumentImpositionColor PrintTicket setting with the colorant name in the profile's colorantTable MUST be performed as a case-sensitive ASCII comparison after trimming leading and trailing whitespace from each string.	15.4		x ^{FP}
M8.25	For gradients, the specified blending color space in the PageBlendColorSpace PrintTicket setting is used only if no gradient stop color values are specified using sRGB or scRGB colors. If any of the gradient stop color values are specified using sRGB or scRGB colors or the consumer does not understand the PageBlendColorSpace PrintTicket setting, the color interpolation mode of the gradient brush MUST be used instead.	15.5		x ^{FP}
	M8.26 This requirement was removed prior to Edition 1 of this specification.			
M8.27	If the PageDeviceColorSpaceUsage is set to MatchToDeviceDefault and the profile specified by the PageDeviceColorSpaceURI PrintTicket	15.6		x ^P

	setting cannot be used as a device color space profile, elements using the profile MUST be color managed like any other element using a color profile.		
M8.28	If the PageBlendColorSpace PrintTicket setting is set to ICCProfile, the profile MUST be an output profile, otherwise it MUST be ignored.	15.6	× ^{FP}
M8.29	Elements using the named color identified by the DocumentImpositionColor PrintTicket setting MUST appear on all color separations.	15.6	× ^{FP}
M8.30	If an ICC profile is not embedded or associated with a raster image or if the embedded or associated profile is not compatible with the pixel format of the image, the default pixel formats for each color space MUST be treated as defined in §15.3.9.	15.3.9	×
M8.31	Channel and tint float values in CMYK, N-Channel, and Named Color syntax MUST be clamped to the valid range from 0.0 to 1.0 before further processing. If the value is used as input for an ICC profile color transformation, it MUST subsequently be linearly scaled to the range from 0 to 255 or from 0 to 65535, depending on whether the profile uses 8-bit or 16-bit input tables.	15.2.3, 15.2.4, 15.2.5	×

1 I.8.2 SHOULD Conformance Requirements

2 Table I-17. Color SHOULD conformance requirements

ID	Rule	Reference	Producer	Consumer
S8.1	ICC profiles SHOULD be used when embedded in any image format with any color space. Images with integer pixel formats are assumed to have sRGB as the default color space and images with floating point pixel formats are assumed to have scRGB as the default color space; in these cases, an ICC profile is unnecessary.	15.1.8		×
S8.2	If consistency of appearance is important, instead of attaching or embedding a gray ICC profile, the producer SHOULD adjust the gray tone response curve of a grayscale image before adding it to the XPS Document.	15.1.8	×	
S8.3	A producer of XPS documents containing named colors SHOULD create the color profile in such a way that a linear ramp of the channel values corresponding to a named colorant maps to PCS values resulting in the same color appearance for consumers unaware of named colors (or the specific colorant).	15.2.5	×	
S8.4	The ContextColor syntax requires a minimum of 1 Alpha value and 3 Channel values for named colors. It is RECOMMENDED that a 1 or 2 tone profile uses the first 1 or 2 channels, respectively, and specifies 0 for the remaining channels.	15.2.5	×	
S8.5	If the consumer does not know ALL of the colorants named in the clrt tag, it SHOULD treat the profile as if it were a regular N-channel source profile and SHOULD NOT attempt to use any of the known colorants, as that would result in undefined results.	15.2.5		×

S8.6	Support for JPEG CMYK images varies by implementation and SHOULD NOT be used in XPS Documents.	15.3.4.3	x
S8.7	For consumers that do perform separation, the imposition named color is an indicator that the tint level supplied SHOULD be used for all device colorants.	15.4	x ^P
S8.8	Producers SHOULD create the profile used by the imposition color in such a way that it does not lay down excessive ink when printed on a device that does not perform separation.	15.4	x ^P
S8.9	If a consumer understands the PageBlendColorSpace PrintTicket setting, it SHOULD convert all color to the specified blending color space before performing a blend operation.	15.5	x ^P
S8.10	If the PageDeviceColorSpaceUsage PrintTicket setting is set to MatchToDeviceDefault, the device's internal color profile SHOULD be used for color management of all elements not using the profile specified by the PageDeviceColorSpaceProfileURI PrintTicket setting.	15.6	x ^P
S8.11	If the PageDeviceColorSpaceUsage PrintTicket setting is set to OverrideDeviceDefault and the profile specified by the PageDeviceColorSpaceProfileURI PrintTicket setting has a number of channels matching the number of primaries of the device, it SHOULD be used instead of the device's internal color management for all elements.	15.6	x ^P
S8.12	If the PageBlendColorSpace PrintTicket setting is set to ICCProfile, the Uri property of the option specifies an ICC profile defining the color space that SHOULD be used for blending.	15.6	x ^P
S8.13	The PageICMRenderingIntent PrintTicket setting SHOULD be ignored for elements using a profile that specifies the rendering intent in the profile.	15.6	x ^P
S8.14	A consumer incapable of supporting named colors SHOULD treat the colorant table for named colors tag in an ICC profile as a user-defined custom tag, and therefore ignore it. The consumer SHOULD instead use the color tables as provided in the profile to convert the specified colors to the Profile Connection Space (PCS).	15.1.8	x
S8.15	Producers SHOULD restrict ICC profiles to conform to the requirements of the older ICC Color Profile specification, ICC.1:2001-04, when consumer support of the newer ISO version cannot be ascertained.	15.1.8	xU v

1 **I.8.3 OPTIONAL Conformance Requirements**

2 *Table I-18. Color OPTIONAL conformance requirements*

ID	Rule	Reference	Producer	Consumer
O8.1	Consumers are not required to handle all color spaces natively through every processing stage , but, rather, MAY convert data specified in a rich color space other than sRGB to sRGB at an early stage (possibly resulting in reduced fidelity).	15.1		x

O8.2	An ICC profile MAY contain the private tag, "MS00", which specifies an embedded Windows Color System (WCS) profile.	15.1.8	x	x
O8.3	When a named color is used in a gradient brush or with transparency, the result produced by consumers that are not named-color aware MAY differ significantly from the result produced by consumers that are named-color aware.	15.2.5		x
O8.4	A named color profile MAY be used with images for spot coloring.	15.2.5	x	x
O8.5	Consumers MAY perform color separation, if desired.	15.4		x
O8.6	Consumers MAY support alpha and gradient blending with rich color spaces such as sRGB or CMYK. Consumers that encounter any document using non-sRGB colors MAY process those colors using conversion to the simpler sRGB color space, resulting in deviations, especially for alpha blending.	15.5		x
O8.7	If the PageColorManagement PrintTicket setting specifies a value of Driver, the driver MAY color manage elements or convert them to different color spaces.	15.6		x ^P
O8.8	Elements using the profile specified by PageBlendColorSpace PrintTicket setting with a value of ICCProfile MAY be blended naively (channel-by-channel) without converting through PCS.	15.6		x ^P
O8.9	XPS producers and consumers MAY provide color management using ICC profiles conforming to the requirements of ISO 15076-1, "Image technology colour management — Architecture, profile format, and data structure — Part 1: Based on ICC.1:2004-10" [O8.9].	15.1.8	x ^U	v

1 I.9 Document Structure and Interactivity

2 I.9.1 MUST Conformance Requirements

3 Table I-19. Document structure MUST conformance requirements

ID	Rule	Reference	Producer	Consumer
M9.1	In order to merge the table cells and rows correctly, producers MUST specify empty <TableCellStructure> elements for cells that do not break across story fragments.	16.1.2	x ^D	
M9.2	If hyperlinked <Path> or <Glyphs> elements are rendered as overlapping on the page, consumers MUST treat the topmost element as the only hyperlink that can be activated in the overlapping region.	16.2.1		x ^H
M9.3	If a producer specifies a FixedPage.NavigateUri attribute on a <Canvas> element, consumers MUST treat all child elements of that canvas that do not override this value with their own FixedPage.NavigateUri attribute setting as having an associated hyperlink.	16.2.1		x ^H
M9.4	Relative internal hyperlinks between FixedPage parts MUST specify, at a minimum, the named address relative to the FixedDocument part.	16.2.1	x ^H	v ^H
M9.5	In order to be addressable by either a hyperlink or the document outline, the named address MUST appear in the	16.2.1	x ^H	

	<PageContent.LinkTargets> element in the fixed document.			
M9.6	If a named address appears in the <PageContent.LinkTargets> element in the fixed document but is not found in the Name attribute of an element within the associated fixed page, consumers MUST treat the top of the associated fixed page as the named address.	16.2.1	x ^H	
M9.7	If a named address in a URI fragment is not found, consumers MUST ignore the fragment portion of the URI.	16.2.1	x ^H	
M9.8	Internal references MUST specify a page address relative to the fixed document sequence.	16.2.2	x ^H	v ^H
M9.9	Consumers MUST expose every element of the fixed page markup to an accessibility interface in the determined reading order, even if the elements are not referenced in the content structure markup.	16.4.1	x	
M9.10	The Name attribute MUST NOT be specified on any children of a <ResourceDictionary> element.	16.2.3	x	v

1 I.9.2 SHOULD Conformance Requirements

2 Table I-20. Document structure SHOULD conformance requirements

ID	Rule	Reference	Producer	Consumer
S9.1	Every meaningful element in the fixed page markup SHOULD specify a Name attribute in order for the document structure markup to refer to it	16.1.1	x	
S9.2	This requirement was removed prior to Edition 1 of this specification.			
S9.3	Document structure markup SHOULD NOT refer to a single named element more than once in the document content or to a named element that embeds another named element that it also refers to. When referring to a <Canvas> element, producers SHOULD consider all descendant elements to be referenced in markup order.	16.1.1	x ^D	
S9.4	If a <StoryBreak> element is not present at the beginning of the content structure markup, consumers SHOULD consider the markup a continuation of the previous story fragment that must be merged. Likewise, if a <StoryBreak> element is not present at the end of the content structure markup, consumers SHOULD consider the markup a continuation to the next story fragment that must be merged to determine the cross-fragment content structure.	16.1.2		x ^D
S9.5	Producers authoring document structure information SHOULD reference every element of the fixed page markup that has semantic meaning (such as text or images) in the StoryFragments parts.	16.1.2.2	x ^D	
S9.6	If consumers enable user interactivity, they SHOULD support hyperlink activation and addressing.	16.2		x ^H
S9.7	When activating a hyperlink, consumers SHOULD load the specified resource if they understand the URI type. If the URI is an internal reference to the XPS Document, consumers SHOULD navigate to the URI.	16.2.1		x ^H

S9.8	The value of the Name attribute on a <FixedPage>, <Canvas>, <Path>, or <Glyphs> element SHOULD be unique within the scope of the fixed document.	16.2.1	x	v
S9.9	It is RECOMMENDED that Name attribute values on <FixedPage>, <Canvas>, <Path>, and <Glyphs> elements be unique within an entire fixed document sequence.	16.2.1	x	v
S9.10	If the Name attribute is specified, producers SHOULD also create a corresponding <LinkTarget> element in the FixedDocument part within the <PageContent> element that links to the parent fixed page	16.2.3	x	
S9.11	A hyperlink destination in the same fixed document SHOULD be expressed as a relative URI.	16.2.4	x ^H	
S9.12	If selection is supported, consumers SHOULD provide a visual cue over or around selected elements.	16.3		x ^S
S9.13	Selection order within an XPS Document SHOULD follow reading order.	16.3		x ^S
S9.14	In the absence of document structure provided in the XPS Document, consumers SHOULD, at minimum, rely on the markup order to determine reading order.	16.4.1	x	
S9.15	Producers SHOULD order the markup in FixedPage parts to reflect the order in which it is intended to be read.	16.4.1	x	
S9.16	When document structure information is present, consumers SHOULD rely on the order of appearance of named elements in the content structure markup to determine reading order.	16.4.1	x	
S9.17	The RECOMMENDED reading order of a page-centric application is 1) order the content by page, 2) order by story fragment within the page based on the order the <StoryFragment> elements are specified in the StoryFragments part for that page, 3) order by <NamedElement> reference within the <StoryFragment> element, 4) append all un-referenced elements that appear in the fixed page markup, ordered by markup order.	16.4.1	x	
S9.18	Producers SHOULD order <StoryFragment> elements in each StoryFragments part in their intended reading order.	16.4.1	x ^D	
S9.19	The RECOMMENDED reading order of a story-centric application is as follows: 1) Order content by story in the sequence the <Story> elements appear in the DocumentStructure part. 2) Within a story, order <StoryFragmentReference> elements in the sequence they appear in the DocumentStructure part. 3) Within a story fragment, order by <NamedElement> references in the StoryFragments part markup. 4) Append all un-referenced elements that appear in the fixed page markup, ordered by page number, then markup order	16.4.1		x ^D
S9.20	Producers SHOULD order <Story> elements in the DocumentStructure part in their intended reading order.	16.4.1	x ^D	
S9.21	Producers SHOULD order <StoryFragmentReference> elements within a <Story> element in their intended reading order.	16.4.1	x ^D	
S9.22	A screen reader consumer SHOULD read the document according to its reading order.	16.4.2	x	

S9.23	A screen reader SHOULD use the UnicodeString attribute of each <Glyphs> element to determine the text to read.	16.4.2	x
S9.24	If a screen reader provides features to navigate the document by structural elements, such as paragraphs or table rows, it SHOULD use any document structure information included in the XPS Document.	16.4.2	x ^D
S9.25	If the screen reader provides features to describe images, it SHOULD read the text provided in the AutomationProperties.Name and AutomationProperties.HelpText attributes.	16.4.2	x
S9.26	If the screen reader provides features to describe hyperlink addresses, it SHOULD read the text provided in the FixedPage.NavigateUri attribute.	16.4.2	x
S9.27	Images and graphics SHOULD specify text alternatives for images and graphics to make this content accessible to vision-impaired individuals. The AutomationProperties.Name attribute SHOULD contain a short description of the basic contents of the image or vector graphic. Individual <Path> elements that do not provide any semantic meaning (such as a line between sections or outlining a table) SHOULD NOT specify these text alternative attributes.	16.4.3	x
S9.28	An image SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText attributes on the <Path> element that is filled with an <ImageBrush> that describes the content specified by the ImageSource attribute of the <ImageBrush> element.	16.4.3	x
S9.29	A vector graphic (a collection of one or more <Path> elements representing a single drawing) SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText attributes only once, directly on a <Canvas> element wrapping the <Path> elements comprising the graphic.	16.4.3	x
S9.30	Children of <VisualBrush> elements SHOULD NOT be referenced by document structure markup.	16.1.2.13	x ^D

1 I.9.3 OPTIONAL Conformance Requirements

2 Table I-21. Document structure OPTIONAL conformance requirements

ID	Rule	Reference	Producer	Consumer
O9.1	Producers MAY choose to add document structure information to XPS Documents. Consumers MAY ignore any authored document structure or hyperlinks.	Clause 16	x ^D	x ^D
O9.2	Producers MAY provide either the document outline or the document content, or both; consumers MAY ignore either or both.	16.1	x ^D	
O9.3	Consumers MAY choose to interpret document structure markup that refers to a single named element more than once, or refers to a named element that embeds another named element that is also referenced, as duplicate content.	16.1.1		x ^D

09.4	Consumers MAY first attempt to locate named elements for document structure directly from the FixedDocument part markup, where they might appear as <LinkTarget> elements if that named element is also intended as an addressable location.	16.1.1	x ^D
09.5	A <TableStructure> element is the complete definition of a table. An implementation MAY use it to build special functionality, such as row or column selection.	16.1.2.6	x ^D
09.6	Internal hyperlinks can specify a named element fragment relative to a particular fixed document, but consumers MAY interpret such a URI relative to the entire fixed document sequence instead	16.2.1	x ^H
09.7	Consumers MAY ignore the Name attribute.	16.2.3	x
09.8	Consumers MAY ignore the FixedPage.NavigateUri attribute.	16.2.4	x
09.9	Viewing consumers that support interactivity MAY support selection and copying.	16.3	x ^S
09.10	Consumers MAY use the FragmentType attribute of the <StoryFragment> element to determine selection behavior, such as disallowing selection of both the page header and the page contents while allowing independent selection within those stories.	16.3	x ^{DS}
09.11	In the absence of document structure information provided in the XPS Document, consumers MAY infer the reading order from the position of elements on the page.	16.4.1	x
09.12	Consumers MAY use the FragmentType attribute of the <StoryFragment> element to determine reading order by interpreting elements that have FragmentType values of Header and Footer as belonging first or last in the reading order, respectively.	16.4.1	x ^D
09.13	Screen readers MAY inspect the Indices attribute to resolve potential ambiguities in the UnicodeString attribute.	16.4.2	x

1 I.10 XPS Document Package Features

2 I.10.1 MUST Conformance Requirements

3 Table I-22. XPS Document package feature MUST conformance requirements

ID	Rule	Reference	Producer	Consumer
M10.1	Consumers MUST be prepared to correctly process interleaved packages in which the PrintTicket or the portion of the relationship data attaching the PrintTicket appears in the package after the affected part.	17.1		x ^P
M10.2	Consumers MUST be able to consume packages regardless of their interleaving structure.	17.1.3		x
M10.3	Consumers that lack the resources to process a part MUST indicate an error condition.	17.1.3		x

M10.4	When consuming interleaved packages, consumers MUST NOT discard any parts without instruction from a DiscardControl part unless they have the ability to access the parts again.	17.1.3	x	
M10.5	If a consumer encounters a reference to an unknown part, it MUST continue to receive further bytes of the package until the unknown part has been transmitted <i>or</i> until the end of the package is reached (indicating an error condition).	17.1.3	x	
M10.6	The DiscardControl part MUST NOT reference itself.	17.1.4.1	x	v
M10.7	If either the Target attribute or the SentinelPage attribute of the <Discard> element contain an invalid reference (refer outside the package), the <Discard> element MUST be ignored.	17.1.4.1.2	x	
M10.8	All producers and consumers signing and verifying signatures for end users or applications MUST adhere to the XPS Document signature policy, and producers and consumers MUST interpret digital signatures consistently.	17.2.1	x ^A	x ^A
M10.9	Consumers MUST NOT prevent an end user from taking an action solely because doing so will invalidate a signature.	17.2.1	x ^A	
M10.10	An XPS Document MUST be considered signed according to the XPS Document signing policy, regardless of the validity of that signature, if the signing rules described in §17.2.1.1 are observed.	17.2.1.1	x ^A	
M10.11	An XPS Document MUST NOT be considered signed according to the XPS Document signing policy if any part not covered by the signing rules is included in the signature or if any relationship not covered by the signing rules is included in the signature.	17.2.1.1	x ^A	
M10.12	An XPS Document digital signer MUST NOT sign an XPS Document that contains content (parts or relationships parts) to be signed that defines the Markup Compatibility namespace but the signer does not fully understand all elements, attributes, and alternate content representations introduced through the markup compatibility mechanisms.	17.2.1.1	x ^A	
M10.13	An XPS Document digital signature MUST be shown as an incompliant digital signature if it violates any of the signing rules regarding parts or relationships that MUST or MUST NOT be signed.	17.2.1.2	x ^A	
M10.14	An XPS Document digital signature MUST be shown as a broken digital signature if it is not an incompliant digital signature, but the signature fails the signature validation routines described in the OPC.	17.2.1.2	x ^A	
M10.15	An XPS Document digital signature MUST be shown as a questionable digital signature if it is not an incompliant or broken digital signature, but the certificate cannot be authenticated against the certificate authority or the signed content (parts and relationships) contain elements or attributes from an unknown namespace introduced through Markup Compatibility mechanisms.	17.2.1.2	x ^A	
M10.16	An XPS Document digital signature MUST be shown as a valid digital signature if it is not an incompliant, broken, or questionable digital signature.	17.2.1.2	x ^A	

M10.17	To prohibit additional signatures in an XPS Document, the signing application MUST sign all the Digital Signature Origin part's relationships of relationship type Digital Signature with the same signature as the rest of the content.	17.2.1.3	x ^A	
M10.18	XPS Document signatures MUST NOT refer to a remote certificate store. All certificates MUST be stored in the XPS Document either as a Certificate part or in the Digital Signature XML Signature part.	17.2.1.4	x ^A	v ^A
M10.19	To link a <SignatureDefinition> to a signature, the value of the SpotID MUST be specified in the Id attribute of the corresponding <Signature> element in the Digital Signature XML Signature part.	17.2.2.2.1	x ^A	
M10.20	Due to space and rendering limitations, producers MUST NOT assume that consumers will use the values specified in the <SpotLocation> element.	17.2.2.3	x ^A	
M10.21	Consumers MUST display the full value of the <Intent> element to the signing party, either in the signature spot or through some other mechanism.	17.2.2.4		x ^A
M10.22	If specified, the <SignBy> date and time MUST be specified as a complete date plus hours, minutes, and seconds in UTC time, as described in the W3C Note "Date and Time Formats."	17.2.2.5	x ^A	
M10.23	There MUST NOT be more than one DiscardControl package relationship.	17.1.4.1	x ^R	v ^R
M10.24	In some cases, producers might rewrite the contents of a package so that parts are provided more than once, allowing consumers to discard a part in order to free resources for additional processing. Each instance of a part MUST be stored as a new, uniquely named part in the package.	17.1.4.1	x ^R	v ^R

1 I.10.2 SHOULD Conformance Requirements

2 Table I-23. XPS Document package feature SHOULD conformance requirements

ID	Rule	Reference	Producer	Consumer
S10.1	When interleaving, the Content Types stream SHOULD be interleaved according to the recommendations in the OPC specification.	17.1	x	
S10.2	When interleaving, PrintTicket parts SHOULD be written to the package before the part to which they are attached.	17.1	x	
S10.3	When interleaving, the portion of the relationship data attaching the PrintTicket to a part SHOULD be written to the package before the part to which it is attached or in close proximity to the part to which it is attached.	17.1	x	
S10.4	When interleaving, if no PrintTicket settings are specified for a FixedDocumentSequence, FixedDocument, or FixedPage part, an empty PrintTicket part SHOULD be attached to the part, and the portion of the relationship data attaching the empty PrintTicket	17.1	x	

	SHOULD be written to the package before the part to which it is attached or in close proximity to the part to which it is attached.		
S10.5	When interleaving, the last piece of the Relationships part for a FixedPage part SHOULD be written to the package in close proximity to the first piece of the FixedPage part.	17.1	×
S10.6	It is RECOMMENDED that one empty PrintTicket be shared for all parts that attach an empty PrintTicket.	17.1.1	×
S10.7	Producers, such as drivers, that target resource-constrained consumers SHOULD: 1) Conservatively model the memory usage of the device. 2) Interleave pieces of parts in the correct order. 3) Decide when certain parts can be discarded by the consumer and inform the consumer within the package stream. 4) Add to the package a uniquely named copy of a resource that could have been discarded, if the resource is referenced by a part sent later in the stream.	17.1.4	×
S10.8	DiscardControl parts that are not well-formed SHOULD NOT be processed and an error SHOULD NOT be reported.	17.1.4.1	× ^U
S10.9	If a <Discard> element is encountered where either or both of the Target attribute and SentinelPage attribute identify a part which has not been processed yet (is still unknown), the <Discard> element SHOULD be retained until both parts identified by the Target attribute and SentinelPage attribute have been processed or until the end of the package is reached.	17.1.4.1.2	× ^U
S10.10	When adding a digital signature to an interleaved package, producers of digitally signed documents that are intended for streaming consumption SHOULD add all digital signature parts and the package relationship to the digital signature parts at the beginning of the package, before adding any other part.	17.1.5	× ^A
S10.11	Consumers SHOULD inform the end user if an action they are going to take will invalidate an existing signature.	17.2.1	× ^A
S10.12	When printing signed documents, the PrintTicket setting JobDigitalSignatureProcessing SHOULD be used to control the digital signature processing behavior. Consumers SHOULD process this PrintTicket setting, if present	17.2.1.5	× ^{AP}
S10.13	If the location specified by the <SpotLocation> element is not used when the signature spot is displayed, it is RECOMMENDED that consumers choose a location that does not contain any page content.	17.2.2.3	× ^A
S10.14	It is RECOMMENDED that consumers render signature spots as consistently sized rectangles that include the signer name, the intent, the signing location, and the scope of the XPS Document to be signed.	17.2.2.3	× ^A
S10.15	It is RECOMMENDED that a signature spot be a clickable area used to launch the digital signing process.	17.2.2.3	× ^A
S10.16	If the <SignBy> element is specified, the consumer SHOULD NOT allow the signing party to sign the document using this particular signature spot after the date and time specified.	17.2.2.5	× ^A

S10.17	The values specified in the Core Properties part SHOULD refer to the entire fixed payload, including the root FixedDocumentSequence part and the compilation of all FixedDocument parts it references.	17.3	x ^C
S10.18	Head-first XPS Document consumers SHOULD attempt to detect inconsistent packages as soon as possible and SHOULD generate an error message, even if they have already processed the pages that resulted in the error.	17.1	x ^K
S10.19	The viewing consumer SHOULD use the values specified in the <SpotLocation> element to place a signature spot.	17.2.2.3	x ^F

1 I.10.3 OPTIONAL Conformance Requirements

2 Table I-24. XPS Document package feature OPTIONAL conformance requirements

ID	Rule	Reference	Producer	Consumer
O10.1	Interleaving is OPTIONAL.	17.1	x	
O10.2	Producers MAY optimize the interleaving order of parts to help consumers avoid stalls during read-time streaming, and to allow consumers to manage their memory resources more efficiently.	17.1.2	x	
O10.3	Consumers MAY discard FixedPage parts once they have been processed.	17.1.3		x
O10.4	Consumers MAY discard FixedDocument and FixedDocumentSequence parts after all their child elements and their closing tags have been processed.	17.1.3		x
O10.5	In the absence of explicit directives to the contrary, consumers MAY discard parts as directed by the DiscardControl part.	17.1.3		x
O10.6	Some producers (typically drivers) MAY choose a suitable interleaving order by modeling the resource management behavior of the consumer.	17.1.4	x	
O10.7	A consumer MAY decide to ignore a malformed DiscardControl part in its entirety or from the first malformed node onward.	17.1.4.1		x
O10.8	An XPS Document digital signer MAY choose not to sign any content (parts or relationships parts) that defines the Markup Compatibility namespace, even if the content is fully understood.	17.2.1.1	x ^A	
O10.9	An XPS Document digital signature MAY be shown as a questionable digital signature if it is not an incompliant or broken digital signature, but contains some other detectable problem at the discretion of the consumer.	17.2.1.2		x ^A
O10.10	XPS Documents MAY be signed more than once.	17.2.1.3	x ^A	
O10.11	Producers MAY include the JobDigitalSignatureProcessing setting in the job-level PrintTicket within the XPS Document content.	17.2.1.5	x ^A	
O10.12	The SpotID attribute of the <SignatureDefinition> element MAY be used to link to an existing signature.	17.2.2.2.1	x ^A	

O10.13	Consumers MAY choose a size and shape to display a signature spot based on the desired display information and page content.	17.2.2.3	x ^A
O10.14	The <SigningLocation> element MAY be set by the original producer of the XPS Document or by the signing party at the time of signing.	17.2.2.6	x ^A

1 **I.11 Rendering Rules**

2 **I.11.1 MUST Conformance Requirements**

3 *Table I-25. Rendering rules MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M11.1	Producers MUST generate XPS Documents that can be accurately rendered by following the rules described in the "Rendering Rules" clause. Consumers MUST adhere to the rules described in the "Rendering Rules" clause when rendering XPS Documents	Clause 18	x	x
M11.2	If a non-invertible transform is encountered during rendering, consumers MUST omit rendering the affected element and all of its child and descendant elements.	18.1.3		x
M11.3	If a non-invertible transform is encountered on a geometry (as specified directly on the geometry or through concatenation), the geometry MUST be considered to contain no area.	18.1.3		x
M11.4	Producers MUST NOT assume a specific placement error for curve decomposition or rely on side-effects of a specific consumer implementation.	18.1.5	x	
M11.5	Encountering markup with characteristics outside of the consumer-specific implementation limits MUST cause an error condition.	18.2		x
M11.6	The alpha information in TIFF images using an ExtraSamples tag value of 1 and in Windows Media Photo images using pixel formats WICPixelFormat32bppPBGRA, WICPixelFormat64bppPRGBA or WICPixelFormat128bppPRGBAFloat MUST be interpreted as pre-multiplied alpha information.	18.4.1		x
M11.7	Composition MUST have the same effect as the application of the rules in §18.5, in sequence.	18.5		x
M11.8	The precise source coordinates as specified by the viewbox MUST be used to place an up-sampled image tile, which is equivalent to using fractional pixels of the original source image.	18.7.2		x
M11.9	Consumers MUST precisely position the tiles specified by the image brush and visual brush. If the specified values result in fractional device pixels, the consumer MUST calculate a running placement-error delta and adjust the placement of the next tile where the delta reaches a full device pixel in order to keep the tiles from being increasingly out of phase as the expanse of the path is filled.	18.7.3		x

1 **I.11.2 SHOULD Conformance Requirements**2 *Table I-26. Rendering rules SHOULD conformance requirements*

ID	Rule	Reference	Producer	Consumer
S11.1	Coordinates are real numbers. All computations on coordinate values SHOULD be performed with at least single floating-point precision. Final conversion (after all transforms have been computed) to device coordinates SHOULD retain at least as much fractional precision as a 28.4 fixed-point representation before performing pixel coverage calculations.	18.1.2, Table 18-1	x	x
S11.2	An <i>ideal</i> consumer implementation SHOULD render pixels in an 8x8 sub-pixel space, perform an 8x8 box filter sampling, and set the pixel to the resulting color value.	18.1.4		x
S11.3	When rendering a shape, a <i>practical</i> implementation (such as a bi-tonal printing device) SHOULD turn on each pixel whose center (at $x+0.5$) is covered by the shape, or is touched by the shape with the shape extending beyond the pixel center in the positive x or y direction of the device.	18.1.4		x
S11.4	When rendering geometries, consumers SHOULD render curves so they appear smooth from a normal viewing distance.	18.1.5		x
S11.5	When no anti-aliasing is used, abutting shapes that share the same device coordinates for the end-points and control-points of an edge SHOULD be rendered without overlap and without gaps. Ideally, an implementation SHOULD also follow this rule for shapes that are mathematically abutting without sharing device coordinates for end-points and control-points of edges.	18.1.7		x
S11.6	Clipping occurs as if a mask were created from the clip geometry according to the pixel inclusion rules. An ideal consumer SHOULD create such a mask in an 8x8 sub-pixel space and subsequently draw only those sub-pixels of a shape that correspond to "ON" sub-pixels in the mask.	18.1.8		x
S11.7	A practical implementation (such as a bi-tonal printing device) SHOULD create a pixel mask according to the pixel inclusion rules and subsequently draw only those pixels of a shape that correspond to "ON" pixels in the mask. In creating the mask and drawing the shape, the abutment of shapes rule SHOULD be observed so that no pixel of the shape is drawn that would not have been drawn if the clip geometry were another abutting shape.	18.1.8		x
S11.8	A typical consumer SHOULD be able to process markup with the implementation limit characteristics indicated in Table 18-1. Producers SHOULD produce only XPS	18.2		x

Documents that stay within these implementation limits.			
S11.9	This requirement was removed prior to Edition 1 of this specification; its description is retained here for historical purposes. Coordinates are real numbers and SHOULD be computed with at least single floating point precision.	11.2	x
S11.10	If the nesting level of <VisualBrush> elements is higher than 16, a consumer SHOULD attempt to flatten the nested content to a bitmap representation rather than failing to draw.	18.2	x
S11.11	Gradients SHOULD be rendered according to the guidelines described in §18.3.	18.3	x
S11.12	Consumers SHOULD pre-process gradient stops for all gradients using the steps described in §18.3.1.1.	18.3.1.1	x
S11.13	If any gradient stops use an sRGB or scRGB color specification or the consumer does not understand the PageBlendColorSpace PrintTicket setting, consumers SHOULD blend colors between gradient stops in the color space indicated by the ColorInterpolationMode attribute of the gradient brush. If none of the gradient stop elements uses an sRGB or scRGB color specification and the consumer understands the PageBlendColorSpace PrintTicket setting, the PageBlendColorSpace PrintTicket setting SHOULD be used. This SHOULD be a linear, channel-by-channel blend operation.	18.3.1.2	x
S11.14	If a ColorInterpolationMode value of SRgbLinearInterpolation is used, the BLEND() function SHOULD convert the color values to sRGB first, and then perform a linear interpolation between them.	18.3.1.2	x
S11.15	If a ColorInterpolationMode value of ScRgbLinearInterpolation is used, the BLEND() function SHOULD convert the color values to scRGB first, and then perform a linear interpolation between them.	18.3.1.2	x
S11.16	In the presence of transformations or when individual gradient stops are very close, the local color gradient at the offset used in the BLEND() function might be large, resulting in a large change over the extent of a single device pixel. In this case, it is RECOMMENDED that the BLEND() function interpolate the gradient over the extent of each device pixel. Producers SHOULD NOT, however, rely on a specific effect for such dense gradient specifications.	18.3.1.2	x x
S11.17	Producers SHOULD either avoid very close gradient stops to the gradient end point when specifying radial gradients where the outside area is visible or avoid specifying radial gradients with a gradient origin on or outside the ellipse (in which case there is no outside area) to ensure consistent rendering results.	18.3.1.2	x

S11.18	All opacity calculations SHOULD be performed with at least 8-bit precision to provide sufficient quality for nested content.	18.4	×
S11.19	When composing superluminous colors, management of out-of-gamut colors SHOULD be deferred until the result is rendered to the final target, at which point out-of-gamut colors are clipped or color managed.	18.4.1	×
S11.20	The color and appearance of the surface created to hold drawing content as it is composed SHOULD match the destination color and appearance, typically a solid white background for a fixed page or transparent for a canvas.	18.5	×
S11.21	Contours and dashes SHOULD be rendered so that they have the same appearance as if rendered by sweeping the complete length of the contour or dash with a line segment that is perpendicular to the contour and extends with half its length to each side of the contour. All points covered by the sweep of this perpendicular line are part of the dash or contour.	18.6	×
S11.22	Consumers SHOULD ensure that parallel edges of strokes appear parallel.	18.6.1	×
S11.23	Consumers SHOULD produce a visually consistent appearance of stroke thickness for thin lines, regardless of their orientation or how they fit on the device pixel grid.	18.6.2	×
S11.24	Consumers SHOULD select line and curve drawing algorithms that behave symmetrically and result in the same set of device pixels being drawn regardless of the direction of the line or curve (start point and end point exchanged).	18.6.3	×
S11.25	If the current render transform is an invertible matrix, consumers SHOULD perform computations on poly line segments and poly Bézier segments with sufficient accuracy to avoid producing zero-length segments.	18.6.8	×
S11.26	If both width and height of a tile are nearly zero, implementations SHOULD average the color values of the brush contents, resulting in a constant-color brush.	18.7.1	×
S11.27	Producers SHOULD avoid producing extreme cases where either the height, width, or both height and width are nearly zero and SHOULD NOT rely on any specific behavior when they do	18.7.1	×
S11.28	Source sampling SHOULD be done from the center of the pixel and should be mapped to the center of the pixel in the device-space. With one extent of the viewbox zero, sampling SHOULD be done along a line parallel to the non-zero side. With both extents of the viewbox zero, a point sample SHOULD be taken.	18.7.2	×
S11.29	When up-sampling an image presented at a lower resolution than the device resolution, bilinear filtering SHOULD be used.	18.7.2	×

S11.30	When down-sampling an image presented at a higher resolution than the device resolution, at least a bilinear filter SHOULD be used.	18.7.2	x
S11.31	A stroke using the consistent nominal stroke width convention SHOULD be rendered with a width consistent with other strokes using the convention that have the same StrokeThickness attribute value, and consumers aware of this convention SHOULD render such a stroke no thinner than the thinnest visible line that consumer supports without dropouts.		x ^F
S11.32	Producers SHOULD NOT create files containing the extreme degenerate case of StrokeDashArray = "0 0". Such lines SHOULD be rendered as a solid line.	18.6.4.6	x x

1 **I.11.3 OPTIONAL Conformance Requirements**

2 *Table I-27. Rendering rules OPTIONAL conformance requirements*

ID	Rule	Reference	Producer	Consumer
O11.1	Very high resolution devices MAY use lower fractional precision than a 28.4 fixed-point representation to represent device coordinates.	18.1.2		x
O11.2	Consumers MAY use different rendering logic as long as it closely approximates the logic of rendering pixels in an 8x8 sub-pixel space, performing an 8x8 box filter sampling, and setting the pixel to the resulting color value.	18.1.4		x
O11.3	Devices MAY use sub-pixel masking.	18.1.4		x
O11.4	An implementation capable of anti-aliasing MAY draw a thin line in a way that blends with the background to varying degrees.	18.1.4		x
O11.5	A bi-tonal implementation on a printer MAY draw thin lines with or without drop-outs, or by applying half-toning, depending on the desired output quality.	18.1.4		x
O11.6	Consumers MAY apply pixel placement rules optimized for character rendering to individual glyphs in a <Glyphs> element.	18.1.6		x
O11.7	Behavior of blending with very close gradient stops MAY vary in an implementation-specific manner (see S11.16).	18.3.1.2		x
O11.8	When a radial gradient origin is on or outside the ellipse, the "outside" area (outside the cone defined by the origin and the ellipse) MAY be filled with an interpolated color value, depending on the resolution.	18.3.1.2		x
O11.9	In certain scenarios (such as when rendering 3D scenes to a bitmap), producers MAY choose to create pre-multiplied bitmap data specifying "superluminous" colors.	18.4.1	x	
O11.10	Consumers MAY handle superluminous colors natively or MAY instead choose to convert pre-multiplied source data containing superluminous colors to non-pre-multiplied data before composition	18.4.1		x

	by ignoring the superluminous portion of each color channel value.		
O11.11	A consumer MAY choose always to initialize the alpha channel of the surface created to hold the drawing content as it is composed to 0.0 (transparent) and the color value to black.	18.5	×
O11.12	When doing page composition, if all elements on a canvas and the canvas itself are opaque (an opacity of 1.0) and parent or ancestor <Canvas> elements are also opaque, the elements MAY be drawn directly to the containing fixed page (or canvas), provided all render transform and clip values are observed	18.5.1	×
O11.13	When doing page composition, if an element is fully transparent (an opacity of 0.0), it MAY be skipped.	18.5.1	×
O11.14	When doing page composition, if a canvas has an opacity of 0.0, it and all of its child and descendant elements MAY be skipped.	18.5.1	×
O11.15	When doing page composition, if a canvas has a Clip property with no contained area, the canvas and all of its child and descendant elements MAY be skipped.	18.5.1	×
O11.16	When doing page composition, a consumer MAY further restrict the size of the temporary surface it creates by the effective extent of the geometry specified by the Clip property of the canvas.	18.5.1	×
O11.17	When doing page composition, a consumer MAY use methods to achieve transparency other than creating a temporary surface. Such methods MAY include planar mapping.	18.5.1	×
O11.21	If only one of the width and height values of a tile is nearly zero, the brush should be constant-colored along lines parallel to the narrow side of the viewport, but implementations MAY differ.	18.7.1	×
O11.22	Consumers MAY choose to implement a more sophisticated algorithm for down-sampling an image presented at a higher resolution than the device resolution, such as a Fant scaler, to prevent aliasing artifacts.	18.7.2	×
O11.23	Consumers MAY choose any technique desired to achieve the requirement to precisely place a tile possibly resulting in fractional device pixel placement, such as linear filtering for seams, stretching of the tile (up-sampling or down-sampling), or pre-computing multiple tiles and adjusting behavior according to how the tiles fit on a grid.	18.7.3	×
O11.24	Temporary work canvases MAY be re-used when tiling transparent brushes.	18.7.4	×
O11.25	Producers MAY generate a <Path> element intended to be treated as having a consistent nominal stroke width by specifying the StrokeDashArray attribute and by specifying the StrokeDashOffset attribute value less than -1.0 times the sum of all the numbers in the StrokeDashArray attribute value.	18.6.12	×
O11.26	If an implementation chooses to draw thin lines, then it MAY choose to draw them with drop outs, following requirement S11.3 in §18.1.4, or as solid rules of 1 pixel thickness.	18.1.4	×

1 **I.12 Additional Conformance Requirements**

2 **I.12.1 MUST Conformance Requirements**

3 *Table I-28. Additional MUST conformance requirements*

ID	Rule	Reference	Producer	Consumer
M12.1	FixedDocument parts MUST be referenced by <DocumentReference> elements within the FixedDocumentSequence part in ascending order. If additional FixedDocument parts are inserted into a fixed document sequence, producers MUST NOT unintentionally change the order of the existing FixedDocument part references.	-	x	
M12.2	A FixedDocument part MUST NOT be referenced more than once by a FixedDocumentSequence part.	-	x	v
M12.3	A FixedPage part MUST NOT be referenced more than once <i>in total</i> , throughout all FixedDocument parts.	-	x	v
M12.4	FixedPage parts MUST be referenced by <PageContent> elements within a fixed document in ascending order. If additional FixedPage parts are inserted into a FixedDocument part, producers MUST NOT unintentionally change the order of the existing FixedPage part references. Documents in languages for which the reading order of pages is back-to-front can be accommodated by adding <PageContent> elements to the FixedDocument in reverse order or by binding the right side of the page.	-	x	
M12.5	Any FixedDocumentSequence, FixedDocument, or FixedPage part that is reachable from the primary fixed payload root or its related parts by relationship or by the Source attribute on a <DocumentReference> or <PageContent> element MUST have no more than one attached PrintTicket part.	-	x	v
M12.6	Every Font part reachable from the primary fixed payload root or its related parts by relationship or by the Source attribute on a <DocumentReference> or <PageContent> element MUST be a valid OpenType font.	-	x	v
M12.7	The content types defined in this specification MUST NOT include parameters. A consumer MUST treat the presence of parameters on these content types as an error when the affected part is accessed.	I.2	x	v

4 **End of informative text.**

1 **J. Bibliography**

2 Independent JPEG Group. <http://www.ijg.org/files/>

3 *A Nonaliasing, Real-Time Spatial Transform Technique*. Fant, Karl M. *IEEE Computer Graphics*
4 *and Applications* 6 (Jan. 1986): 71–80.

5 *OS/2 and Windows Metrics*. Microsoft Corporation. 2001.

6 <http://www.microsoft.com/typography/otspec/os2.htm>

7 *OpenType Font File*. Microsoft Corporation. 2001.

8 <http://www.microsoft.com/typography/otspec/otff.htm>

9 *OpenType Specification, Version 1.4*. Microsoft Corporation. 2004.

10 <http://www.microsoft.com/typography/otspec/default.htm>

11 *Print Schema*. Microsoft Corporation. 2006. [http://windowssdk.msdn.microsoft.com/en-](http://windowssdk.msdn.microsoft.com/en-us/library/default.aspx)

12 [us/library/default.aspx](http://windowssdk.msdn.microsoft.com/en-us/library/default.aspx)

13 ~~*TIFF, Revision 6.0. Adobe Systems Incorporated. 1992.*~~

14 ~~<http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>~~

15 *Windows Color System in Windows Longhorn, WinHEC 2005 Version*. Microsoft Corporation.

16 2005. [http://download.microsoft.com/download/5/D/6/5D6EAF2B-7DDF-476B-93DC-](http://download.microsoft.com/download/5/D/6/5D6EAF2B-7DDF-476B-93DC-7CF0072878E6/WCS.doc)

17 [7CF0072878E6/WCS.doc](http://download.microsoft.com/download/5/D/6/5D6EAF2B-7DDF-476B-93DC-7CF0072878E6/WCS.doc)

18 Windows Media Photo Microsoft Corporation. <http://www.microsoft.com/xps>

1 K. Index

2 In the index that follows, *italic* page numbers are used to indicate illustrations and examples
 3 with illustrations. **Bold** page numbers are used to indicate a primary reference when several
 4 pages are listed. Page ranges are elided. "See" references indicate the primary index location
 5 for that topic, while "See *also*" references indicate related index topics.

6	<hr/>	49	offset	166
7	A	50	specifying	166
8	abbreviated geometry syntax	51	image brush	
9	syntax	52	described	127 , 129
10	accessibility	53	example	129
11	document structure, enabled by	54	image	<i>See</i> image
12	image text alternative, long	55	mentioned	26, 67
13	described	56	source, specifying	128
14	on canvas	57	tile size and placement	<i>See</i> brush, view port
15	on path	58	tile source	<i>See</i> brush, view box
16	image text alternative, short	59	linear gradient brush	
17	described	60	color interpolation mode	152
18	on canvas	61	described	153
19	on path	62	end point	152
20	importance of	63	example	153
21	mentioned	64	gradient stops, specifying	157
22	of text	65	mappng mode	152
23	page elements, requirement to expose all	66	rendering	275
24	reading order	67	specifying	152
25	document structure, dependent on	68	spread method	
26	fragment type, dependent on	69	described	154
27	markup order, dependent on	70	Pad	154
28	of page-centric application	71	Reflect	155
29	of story-centric application	72	Repeat	156
30	screen reader	73	specifying	152
31	considerations	74	start point	152
32	mentioned	75	opacity	<i>See</i> opacity
33	alpha	76	radial gradient brush	
34	anti-aliasing	77	color interpolation mode	158
35	disabling of	78	described	160
36	rendering of	79	example	161
37	application media size	80	gradient center	158
38	arc	81	gradient origin	158
		82	gradient stops, specifying	165
		83	gradient x-radius	158
		84	gradient y-radius	158
		85	mappng mode	158
39	B	86	rendering	277
40	bleed area	87	specifying	158
41	bookmark	88	spread method	
42	brush	89	described	162
43	alpha	90	Pad	162
44	described	91	Reflect	163
45	gradient computations	92	Repeat	164
46	gradient stop	93	specifying	158
47	color, specifying	94	solid color brush	126, 127
48	described	95	tile	

1 behavior See brush, tile, mode
 2 image scaling 299
 3 mode
 4 described 141–51
 5 FlipX 146–47
 6 FlipXY 150–51
 7 FlipY 148–49
 8 mentioned 136
 9 None 141–43
 10 specifying 128, 131, **136**
 11 Tile 144–45
 12 placement See brush, view port
 13 size See brush, view port
 14 small tile rendering 299
 15 source See brush, view box
 16 transparent brush tiling 300
 17 transformation See transformation
 18 view box
 19 calculating source coordinates for images 136
 20 described 136
 21 example 137–40
 22 larger than image 136, 140
 23 mapping to view port 128, 131
 24 specifying for image brush 128
 25 specifying for visual brush 131
 26 syntax 128, 131
 27 unit type 128, 131, 136
 28 units, for images 128, **136**
 29 view port
 30 described 136
 31 example 137–40
 32 placement precision 299
 33 specifying for image brush 128
 34 specifying for visual brush 131
 35 syntax 128, 131
 36 unit type 128, 131, 136
 37 visual brush
 38 described **131**, 132
 39 example 134
 40 visual 131, 132

41 **C**

42 canvas
 43 anti-aliasing control 61
 44 clipping See clipping
 45 composing properties 63
 46 described 61
 47 opacity See opacity
 48 opacity mask See opacity mask
 49 transformation See transformation
 50 caret stop See selection, caret stop
 51 CFF font See font:CFF
 52 circle See geometry, segment, arc
 53 clipping
 54 described 183

55 elements applied to 171, 172
 56 geometry, reference to 76
 57 of canvas 61, 183
 58 of glyphs 98, 185
 59 of path 69, 184
 60 rules 271
 61 CMYK See color, color space, CMYK
 62 color
 63 alpha See opacity
 64 black generation 224
 65 blending **7**
 66 behavior described 221
 67 blend color space for linear gradient brush 152
 68 blend color space for radial gradient brush 158
 69 color space 223
 70 for gradients 274
 71 implementation dependent 221
 72 brush, specifying for See brush
 73 color management on device 222
 74 color profile
 75 embedded in image 26, 27, 29, 31, 211
 76 ICC profile
 77 color channels supported 209, 210
 78 colorant table usage for named colors 211
 79 described 210
 80 grayscale image, usage with 211
 81 parts See parts
 82 profile types allowed 210
 83 version supported 209
 84 Windows Color System (WCS) profile
 85 embedded in ICC profile 211
 86 mentioned 209, 212
 87 WcsProfilesTag 211
 88 color separation 221
 89 color space
 90 CMYK 210
 91 in images 209
 92 in vector graphics 209
 93 device color 210, 222
 94 gray colors 210
 95 ICC profile
 96 version supported 210, 427
 97 named color **8**, 210
 98 named colors 211
 99 n-channel 210
 100 in images 209
 101 in vector graphics 209
 102 sRGB 210
 103 gamut boundary definition 395
 104 in images 209
 105 in vector graphics 209
 106 spot colors See color, color space, named colors
 107 sRGB 209
 108 in images 209
 109 in vector graphics 209
 110 support required 209
 111 support summarized 209

1	document imposition color	224	55	store	263
2	fidelity, improved	209	56	validity	263
3	gamma	<i>See color, blending</i>	57	co-signature	<i>See digital signature, signature definitions</i>
4	pixel formats	<i>See color, raster image support</i>	58	multiple signatures	263
5	printing	<i>See PrintTicket keywords</i>	59	namespace	<i>See namespace</i>
6	raster image support		60	Open Packaging Conventions, extended from	260
7	associating a color profile part	220	61	origin	
8	CMYK	218	62	part	<i>See parts</i>
9	device color	220	63	relationship to	24
10	gray colors	218	64	parts	<i>See parts</i>
11	named colors	219	65	printing	263
12	n-channel	219	66	relationship to	24
13	pixel format defaults	220	67	relationships	<i>See relationships</i>
14	scRGB	218	68	request	<i>See digital signature, signature definitions</i>
15	sRGB	217	69	signature definitions	9
16	rendering intent, specifying	224	70	described	39, 264
17	syntax		71	markup example	265
18	CMYK	215	72	mentioned	22
19	named color	216	73	namespace	<i>See namespace</i>
20	n-channel	215	74	relationship to	24
21	scRGB	214	75	sign by date and time	268
22	sRGB	213	76	signer name	266
23	summarized	213	77	signing intent	267
24	where used	212	78	signing location	268
25	color profile	<i>See color, color profile</i>	79	specifying	265
26	composability of properties	<i>See XPS Document format,</i>	80	spot ID	266
27	properties, composability		81	spot location	266, 267
28	compression		82	signature policy	
29	image	<i>See image</i>	83	conditions where policy does not apply	262
30	package	<i>See Open Packaging Conventions, specification</i>	84	described	261
31	conformance		85	markup compatibility impact	262, 263
32	inherited from Open Packaging Conventions	17	86	parts to sign	
33	language notes	3	87	optional	261
34	of software	3	88	required	261
35	requirements tables	405–46	89	relationships to sign	
36	consumer	7	90	as a group, required	262
37	described	17	91	conditionally required	262
38	implementation burden	405–46	92	required	262
39	content area	<i>See page, content area</i>	93	signing rules	9 , 261–62
40	content type	7	94	signing validity	262–63
41	namespace	<i>See namespace</i>	95	single signature	262
42	summarized	402	96	signature spot	<i>See digital signature, signature definitions</i>
43	usage of	21	97	signature status	
44	contour intersection point	7	98	broken	7
45	copy and paste	<i>See also selection</i>	99	broken digital signature	263
46	document structure, improved by	227	100	compliant	7
47	core properties	<i>See Open Packaging Conventions</i>	101	incompliant	7
48	curve	<i>See geometry, segment</i>	102	incompliant digital signature	262
			103	questionable	7
			104	questionable digital signature	263
			105	valid	7
			106	valid digital signature	263
49	D		107	discard control	
50	device	7	108	consumer considerations	258
51	device color	<i>See color, color space, device color</i>	109	elements	259
52	digital signature		110	markup example	259
53	certificate		111	namespace	<i>See namespace</i>
54	relationship to	24			

1 part..... See parts

2 reference, invalid260

3 reference, not yet encountered260

4 resource constraints, addressing258

5 sentinel page.....260

6 target resource.....260

7 usage of.....258, 272

8 document

9 markup..... See XPS elements, document-level

10 namespace See namespace

11 order of49

12 reference to49

13 document conventions11

14 diagram notes11

15 document outline See document structure, outline

16 document roll-up..... See document, order of

17 document sequence, namespace See namespace

18 document structure

19 constructed algorithmically.....40

20 constructed explicitly40

21 content.....227

22 content structure7

23 document content.....7

24 figure246

25 list

26 item

27 marker246

28 specifying246

29 specifying245

30 markup example228

31 named element8

32 canvas descendant elements.....228

33 described227

34 link target, optimizing location with228

35 markup compatibility, updating for247

36 referencing each once227

37 specifying247

38 visual brush descendants prohibited228

39 namespace See namespace

40 naming page elements171, 227

41 outline7, 227

42 language See language, of outline , See language, of

43 outline

44 levels.....230

45 markup described229

46 markup example230

47 mentioned40

48 outline entry229

49 target URI.....230

50 paragraph

51 specifying243

52 parts See parts

53 relationships..... See relationships

54 section

55 specifying243

56 story9

57 described40, 227

58 markup231

59 markup example233

60 story fragment, correlating to page232

61 story fragment, reference to232

62 story fragment9

63 break indicator234, 243

64 content structure, contains40, 227

65 described40, 227

66 fragment name.....239

67 fragment name, uniqueness232

68 fragment type239

69 markup described237, 239

70 markup elements summarized.....233

71 markup example240, 241

72 merging fragments234–37

73 referencing every page element240

74 relation to fixed page227

75 story belonged to239

76 story, not belonging to any240

77 story fragments See namespace

78 table

79 cell

80 column span245

81 merging.....234

82 row span245

83 specifying245

84 row group, specifying244

85 row, specifying244

86 specifying244

87 thread See document structure, story

88 usage of40

89 usage optional227

90 driver7

91 described258

92 **E**

93 effective coordinate space See layout, coordinate space

94 elements See XPS elements

95 EXIF

96 usage in JPEG See image, JPEG, EXIF

97 usage in TIFF See image, TIFF, EXIF

98 extensibility of XPS Documents See markup compatibility

99 **F**

100 figure

101 illustration..... See document structure, figure

102 shape See geometry, figure

103 fill

104 algorithm See geometry, fill algorithm

105 of path See path, fill brush

106 of stroke..... See path, stroke brush

107 find47, 111, 119

108 fixed document sequence See document, order of

109 fixed page See page

- 1 fixed payload*See* XPS Document format
- 2 FixedDocument part**8**
- 3 FixedDocumentSequence part.....**8**
- 4 FixedPage part**8**
- 5 font
- 6 CFF.....33, 113
- 7 compatibility encoding36
- 8 device font98, 118
- 9 embedding**33–36**, 35
- 10 extraction34, 35
- 11 language impact on copy and paste.....47
- 12 licensing rights.....33, 35, 36
- 13 obfuscation34, **35**
- 14 algorithm35
- 15 OpenType.....33
- 16 parts*See* parts
- 17 rasterization34
- 18 relationships.....*See* relationships, required resource
- 19 restricted editing..... *See* relationships, restricted font
- 20 restricted font25
- 21 sharing.....33
- 22 subsetting.....33, 34
- 23 TrueType33
- 24 TrueType collection (TTC).....33, 416
- 25 Unicode encoding.....33
- 26 usage of.....33
-
- 27 **G**
- 28 geometry
- 29 abbreviated syntax79, **89–95**
- 30 algorithm389
- 31 circle..... *See* geometry, segment, arc
- 32 curve..... *See* geometry, segment
- 33 described75
- 34 figure
- 35 closed.....**80, 88**, 92
- 36 described75
- 37 fill control.....80
- 38 markup.....80
- 39 reference to77, 79
- 40 segments, composed of75
- 41 start point80, 90
- 42 stroking of segments75
- 43 figures, composed of75
- 44 fill algorithm
- 45 described78
- 46 EvenOdd78
- 47 mentioned75
- 48 NonZero79
- 49 specifying77, 90
- 50 filled area77
- 51 segment
- 52 arc81–84, 91
- 53 Bézier curve85, 91
- 54 Bézier curve, quadratic87, 91
- 55 Bézier curve, smooth..... 91, 93
- 56 line..... 86, 90
- 57 segment, degenerate..... 297
- 58 transformation..... *See* transformation
- 59 usage described 76
- 60 glyphs *See* text
- 61 gradient *See* brush, *See* brush
- 62 graphics *See* path
- 63 grouping markup*See* canvas
-
- 64 **H**
- 65 hairline*See* stroke, hairline
- 66 hyperlink
- 67 activation 247
- 68 addressability
- 69 appearance in link targets 248
- 70 mentioned 51, 248
- 71 missing name, handling of..... 248
- 72 name 248
- 73 name uniqueness 248
- 74 of canvas 61, 248
- 75 of glyphs 98, 248
- 76 of page..... 53, 248
- 77 of path..... 69, 248
- 78 of visual brush contents 248
- 79 page number 248
- 80 document-level listing *See* hyperlink, target
- 81 example 248
- 82 mentioned 227
- 83 overlapping behavior 247
- 84 source
- 85 base URI 249
- 86 described 249
- 87 from canvas..... 61, 247
- 88 from glyphs..... 98, 247
- 89 from path 69, 247
- 90 inheritance of 247
- 91 specifying 171
- 92 support recommended 249
- 93 target
- 94 addressability*See* hyperlink, addressability
- 95 document-level definition of..... 52
- 96 external 247
- 97 internal 247
- 98 link target, specifying as 51
- 99 missing behavior 248
- 100 name*See* name
- 101 relative target handling 248
- 102 relative target recommended 249
- 103 relative to document, at minimum 247
-
- 104 **I**
- 105 ICC *See* color, color profile
- 106 image

1 brush See brush, image brush

2 JPEG..... 22, 26

3 APP markers 26

4 CMYK 27

5 EXIF 26

6 naming 26

7 specification 26

8 parts See parts

9 PNG 22, 26, 27

10 chunks 27

11 naming 27

12 specification 27

13 relationships See relationships, required resource

14 resolution 136

15 resource See resource

16 sharing 26

17 thumbnail 22

18 TIFF 22, 26, 28

19 alpha, associated 31

20 CCITT bilevel encoding 30

21 CMYK 30

22 compression 31

23 EXIF 31

24 features, supported 30

25 image file directory (IFD) 29

26 naming 28

27 specification 28, 31

28 tags, supported 28

29 tags, unsupported 30

30 tags, unsupported 31

31 variations, handling 31

32 types supported 26, 129

33 usage described 26, 67

34 Windows Media Photo 22, 26, 31

35 CMYK 32

36 compression See image, Windows Media Photo,

37 specification

38 features supported 32

39 grayscale 32

40 named color 32

41 naming 31

42 N-channel 32

43 profiled RGB 32

44 scRGB 32

45 specification 31, 447

46 sRGB 32

47 imageable size See PrintTicket keywords, PageImageableSize

48 implementation limits 272–73

49 ink area See page, content area

50 interleaving See Open Packaging Conventions, interleaving

51 **J**

52 JPEG See image, JPEG

53 **L**

54 landscape orientation See page, orientation of

55 language

56 markup 48, 172

57 of canvas 61

58 of glyphs 98, 119

59 of outline 48, 229

60 of outline entry 48, 230

61 of page 53

62 of path 69

63 of resource 177

64 of signature definition 266

65 usage 47

66 layout See also transformation, matrix

67 composition

68 behavior 284

69 examples 285–87

70 optimization 284

71 rules 283

72 coordinate rounding 269

73 coordinate space 186

74 compositability 54

75 effective coordinate space 7

76 described 46

77 mentioned 53, 69, 98, 110, 125, 136, 152, 158, 172,

78 180

79 mentioned 49

80 origin 269

81 transformation See transformation

82 units 269

83 x-axis 269

84 y-axis 269

85 degenerate segments 297

86 implementation limits See implementation limits

87 page dimensions See page

88 pixel

89 center location See layout, pixel

90 inclusion 270

91 placement 270

92 placement behavior for glyphs 271

93 placement error maximum 271

94 rendering 270

95 sub-pixel masking 271

96 PrintTicket interactions 56

97 shape abutment 271

98 line

99 characteristics of See stroke

100 curved See geometry, segment

101 drawing of See path

102 geometry of See geometry, segment, line

103 linear gradient See brush, linear gradient brush

104 link See hyperlink

105 link target See hyperlink, target

106 list See document structure, list

1 M

2 markup compatibility
 3 digital signature, impacted by262
 4 document structure, usage in247
 5 mentioned1
 6 namespace*See namespace*
 7 mentioned45
 8 preprocessing requirements44
 9 processing required44
 10 property elements, usage with46
 11 resource dictionary, usage in182
 12 usage of44
 13 media size *See PrintTicket keywords, PageMedaSize*
 14 media size, application56
 15 memory management, device*See interleaving; discard*
 16 control
 17 miter *See stroke, line, join*

18 N

19 name
 20 elements applied to171
 21 link target correspondence249
 22 link target, specifying as51
 23 purpose of248
 24 resource entries, prohibited for249
 25 syntax249
 26 uniqueness248
 27 named color*See color, color space, named color*
 28 namespace401
 29 content type401
 30 core properties401
 31 digital signatures401
 32 discard control401
 33 document401
 34 document sequence401
 35 document structure401
 36 markup compatibility401
 37 page401
 38 Print Schema framework401
 39 Print Schema keywords401
 40 relationships401
 41 resource dictionary key401
 42 signature definitions401
 43 story fragments401
 44 naming of parts*See parts, naming recommendations*
 45 natural language*See language*
 46 n-channel color *See color, color space, n-channel*

47 O

48 opacity
 49 blending *See color, blending*
 50 brush initial opacity136

51 composition effects284
 52 computations280–82
 53 described172
 54 elements applied to171
 55 mask*See opacity mask*
 56 of canvas61
 57 of color32, 126, 152, 158, 209, 213, 215, 216, 221
 58 of glyphs98
 59 of image brush128
 60 of linear gradient brush152
 61 of path69
 62 of pixel formats217, 218, 219
 63 of radial gradient brush158
 64 of solid color brush126
 65 of stroke284
 66 of visual brush131
 67 pre-multiplied alpha282
 68 superluminous colors282
 69 transparent brush tiling300
 70 value range280
 71 opacity mask
 72 brush, filling with *See brush*
 73 described167
 74 elements applied to171, 172
 75 example167, 169
 76 of canvas61, 203
 77 of glyphs98, 205
 78 of path69, 204
 79 Open Packaging Conventions
 80 ordering
 81 simple **8**
 82 Open Packaging Conventions
 83 ordering
 84 interleaved **8**
 85 Open Packaging Conventions
 86 package **8**
 87 Open Packaging Conventions
 88 package
 89 packaging model **8**
 90 Open Packaging Conventions
 91 package
 92 relationship **8**
 93 Open Packaging Conventions
 94 physical model **8**
 95 Open Packaging Conventions
 96 interleaving
 97 piece **8**
 98 Open Packaging Conventions
 99 physical model19
 100 Open Packaging Conventions
 101 package
 102 packaging model19
 103 Open Packaging Conventions
 104 interleaving
 105 optimization253
 106 Open Packaging Conventions
 107 interleaving

- 1 parsing head-first vs. tail first253
- 2 Open Packaging Conventions
- 3 interleaving
- 4 consumer considerations.....258
- 5 Open Packaging Conventions
- 6 digital signature..... See digital signature
- 7 Open Packaging Conventions
- 8 core properties.....268
- 9 Open Packaging Conventions
- 10 core properties
- 11 namespace..... See namespace
- 12 OpenType font..... See font:OpenType
- 13 optimization
- 14 for streaming consumption..... See Open Packaging
- 15 Conventions, interleaving
- 16 of composition rules284
- 17 of digital signatures260
- 18 of glyphs119
- 19 of interleaving255–58
- 20 of named element location228
- 21 of pixel placement rules271
- 22 outline..... See document structure, outline
-
- 23 **P**
- 24 packaging model..... See Open Packaging Conventions
- 25 page
- 26 bleed area53, 54
- 27 content area53, 55, 269
- 28 height of53, 269
- 29 height, advisory51
- 30 imageable area..... See PrintTicket keywords,
- 31 PageImageableSize
- 32 layout See also layout
- 33 markup grouping See canvas
- 34 namespace See namespace
- 35 order of pages50
- 36 orientation of55, 56, 58
- 37 reference to50
- 38 root of53
- 39 scaling for print56
- 40 size terminology55
- 41 uniqueness of51
- 42 width of53, 269
- 43 width, advisory51
- 44 paragraph See document structure, paragraph
- 45 part8
- 46 part name.....8
- 47 parts..... See also XPS Document format
- 48 core properties..... See Open Packaging Conventions,
- 49 specification
- 50 digital signature
- 51 certificate
- 52 part..... See Open Packaging Conventions, specification
- 53 digital signature origin See Open Packaging Conventions,
- 54 specification
- 55 DiscardControl 22, 258
- 56 DocumentStructure 22, 40, 227
- 57 FixedDocument..... 21, 25
- 58 FixedDocumentSequence 21, 25
- 59 FixedPage..... 21, 26
- 60 font 21, 33–37
- 61 ICC profile 22
- 62 image 22, 26–32
- 63 naming recommendations..... 41–43
- 64 PrintTicket..... 22, 37
- 65 remote resource dictionary 22, 37
- 66 SignatureDefinitions 22, 39
- 67 StoryFragments 22, 40, 233
- 68 thumbnail 22, 32
- 69 thumbnail, package See Open Packaging Conventions,
- 70 specification
- 71 XML digital signature See Open Packaging Conventions,
- 72 specification
- 73 path
- 74 clipping See clipping
- 75 described 67, 68
- 76 fill brush 69
- 77 geometry, reference to 69, 72
- 78 opacity See opacity
- 79 opacity mask..... See opacity mask
- 80 shape See geometry
- 81 stroke brush..... 69
- 82 stroke control See stroke
- 83 transformation..... See transformation
- 84 usage described 72
- 85 payload..... 8, See XPS Document format
- 86 physical imageable size 8, See PrintTicket keywords,
- 87 PageImageableSize
- 88 physical media size 8, See PrintTicket keywords,
- 89 PageMedaSize
- 90 physical model See Open Packaging Conventions
- 91 physical organization..... See Open Packaging Conventions,
- 92 interleaving; ZIP
- 93 pixel See layout, pixel
- 94 pixel snapping See stroke, line
- 95 PNG See image, PNG
- 96 portrait orientation See page, orientation of
- 97 positioning content See layout; transformation, matrix
- 98 primary fixed payload root..... 8
- 99 Print Schema See PrintTicket
- 100 printing
- 101 bleed area See page, bleed area
- 102 content area See page, content area
- 103 device fonts See font, device font
- 104 digital signature See digital signature
- 105 discard control See discard control
- 106 font, print and preview restricted . See font, licensing rights
- 107 interleaving . See Open Packaging Conventions, interleaving
- 108 layout..... See layout
- 109 orientation See PrintTicket keywords
- 110 PrintTicket..... See PrintTicket; PrintTicket keywords
- 111 resource constraints See discard control

1	scaling.....	See PrintTicket keywords	55	StartPart.....	21, 25
2	PrintTicket.....	8	56	StoryFragments.....	25
3	described.....	37	57	thumbnail.....	25
4	empty PrintTicket, markup of.....	255	58	usage of.....	25
5	mapping content levels to parts.....	38	59	relationships part.....	9
6	namespace.....	See namespace	60	rendering.....	See also layout
7	parts.....	See parts	61	rules described.....	269–300
8	processing requirements.....	38	62	required part.....	9
9	relationships.....	See relationships	63	resource.....	See resource dictionary, resource definition
10	PrintTicket keywords		64	resource constraints.....	See discard control
11	DocumentImpositionColor.....	224	65	resource definition.....	See resource dictionary, resource definition
12	ICMRenderingIntent.....	224	66	resource dictionary.....	9
13	JobDigitalSignatureProcessing.....	263	67	described.....	172, 176
14	PrintInvalidSignatures.....	263	68	example.....	173, 175, 177, 178, 180
15	PrintInvalidSignaturesWithErrorReport.....	263	69	markup compatibility usage.....	182
16	PrintOnlyValidSignatures.....	263	70	remote.....	9 , See resource part, remote resource dictionary
17	namespace.....	See namespace	71	resource definition.....	9
18	PageBlackGenerationProcessing.....	224	72	described.....	172, 176
19	PageBlendColorSpace.....	223	73	key	
20	PageColorManagement.....	222	74	described.....	172, 176
21	PageDeviceColorSpaceProfileURI.....	222	75	described.....	172
22	PageDeviceColorSpaceUsage.....	222	76	namespace.....	See namespace
23	PageImageableSize.....	55, 58, 59, 60	77	on canvas.....	61
24	PageMediaSize.....	55, 56, 58, 60, 269	78	on geometry.....	77
25	PageOrientation.....	55, 56, 58	79	on glyphs.....	98
26	PageScaling.....	58–60	80	on image brush.....	128
27	producer.....	9	81	on linear gradient brush.....	152
28	bleed size.....	9 , 55	82	on matrix transformation.....	186
29	content size.....	9 , 55	83	on path.....	69
30	described.....	17	84	on radial gradient brush.....	158
31	implementation burden.....	405–46	85	on solid color brush.....	126
32	media size.....	9 , 55	86	on visual brush.....	131
33	property.....	See XPS Document format, properties	87	uniqueness.....	180
<hr/>					
34	R		88	language.....	177
35	radial gradient.....	See brush, radial gradient brush	89	locating.....	181
36	raster graphics.....	See image	90	namespace prefixes, interpreting.....	177
37	reading order.....	See accessibility, reading order	91	referencing previously-defined resources... 176, 177, 180	
38	relationship.....	9	92	usefulness of path, glyphs, and canvas as.....	173
39	StartPart.....	10	93	sharing.....	See resource part, remote resource dictionary
40	relationships.....	See also XPS Document format	94	specifying.....	172, 173–77
41	core properties.....	24	95	where usable.....	173
42	digital signature.....	24	96	resource part	
43	digital signature certificate.....	24	97	color profile.....	See color, color profile
44	digital signature definitions.....	24	98	font.....	See font
45	digital signature origin.....	24	99	image.....	See image
46	DiscardControl.....	24	100	reference to.....	24, 26
47	DocumentStructure.....	24	101	relationships to.....	See relationships
48	external prohibited.....	24	102	remote resource dictionary.....	22, 24, 173, 177
49	namespace.....	See namespace	103	required resource.....	24
50	PrintTicket.....	24	104	usage of.....	24
51	purpose of.....	24	105	resource reference.....	9
52	relationship types, reference table.....	403	106	described.....	172, 180
53	required resource.....	24	107	example.....	180
54	restricted font.....	25, 34, 36	108	scope.....	180
			109	syntax.....	180
			110	rotating content.....	See transformation, matrix

1 **S**

2 scalable vector graphics (SVG)*See geometry, abbreviated*

3 syntax

4 scaling

5 content.....*See transformation, matrix*

6 for print..... *See page:scaling for print*

7 scRGB..... *See color, color space, scRGB*

8 search*See find*

9 section*See document structure, section*

10 segment *See geometry, segment*

11 selection

12 behavior250

13 caret stop98, 119

14 document structure, enabled by..... 227, 250

15 fragment type, behavior depending on250

16 mentioned..... 111, 119, 227

17 order250

18 recommended250

19 shear*See transformation, matrix*

20 signature definitions *See digital signature*

21 signature spot**9**, *See digital signature, signature definitions*

22 size terminology *See page, size terminology*

23 skewing content*See transformation, matrix*

24 sRGB.....*See color, color space, sRGB*

25 starting part..... *See relationships, StartPart*

26 story..... *See document structure, story*

27 story fragment *See document structure, story fragment*

28 stream.....**10**

29 stretching content*See transformation, matrix*

30 stroke

31 brush*See path, stroke brush*

32 consistent nominal width..... *See stroke, hairline*

33 contour rendering288

34 dash

35 cap289

36 flat289

37 round.....290

38 square289

39 style.....69

40 triangle290

41 offset.....69

42 overlapping.....291

43 style69

44 drawing algorithm, symmetry of.....288

45 edge parallelization288

46 fill rule, independence from.....298

47 hairline298

48 line

49 cap291

50 end69

51 flat292

52 for dashed stroke292

53 round.....292

54 square292

55 start.....69

56 triangle292

57 join

58 bevel 294

59 miter 295–97

60 round 293

61 style 69

62 miter limit..... 69

63 pixel snapping 69

64 thin stroke anti-aliasing behavior 271

65 zero-length, avoided 81

66 multi-figure path, behavior with..... 298

67 phase control..... 288

68 segments, mixed stroked and non-stroked 298

69 thickness..... 69

70 **T**

71 table *See document structure, table, See document structure,*

72 table

73 table of contents *See document structure, outline*

74 text

75 baseline..... *See text, glyph, baseline*

76 bidirectional..... 98, 115

77 bold..... *See text, style simulation*

78 clipping *See clipping*

79 fill brush 98, 124

80 font*See also font*

81 device font, reference to..... 98, 118

82 reference to..... 98

83 glyph

84 advance width103, 104, 109, 110

85 baseline 103

86 black box 103

87 cluster map.....104–8, 109, **110**, 111

88 indices

89 described 109

90 reference to non-existent glyph 109

91 restriction of length 109

92 specifying 98

93 syntax..... 109

94 metrics 103

95 offset 104, 109, 110, 115

96 origin 103, 104

97 origin, sideways..... 103

98 side-bearing, bottom..... 104

99 side-bearing, left 103

100 side-bearing, right 103

101 side-bearing, top 104

102 glyphs usage for 97

103 italic *See text, style simulation*

104 markup 98

105 markup examples 121

106 markup optimization 119

107 opacity *See opacity*

108 opacity mask.....*See opacity mask*

109 position..... *See also text, glyph, offset*

110 horizontal text 98

- 1 vertical text 112
- 2 sideways (vertical) 98
- 3 advance width 109, **113**
- 4 bidirectional text, intersection with 113
- 5 described 112
- 6 example, sideways 116
- 7 example, vertical 116, 117
- 8 horizontal text, including 112
- 9 origin calculation 112
- 10 vertical glyphs, preference for 112
- 11 size 98
- 12 style simulation 98, **111**
- 13 transformation *See transformation*
- 14 underline *See path*
- 15 Unicode string *See also text, glyph*
- 16 escaping open brace character 111
- 17 mapping code units to glyphs 104
- 18 normalization prohibited 111
- 19 specifying 98
- 20 Unicode control marks, inclusion of 111
- 21 Unicode scalar value, split into code units 104
- 22 unmappable code unit behavior 111
- 23 usage of **111**
- 24 UTF-16 code units, consisting of 104
- 25 vertical *See text, sideways (vertical)*
- 26 thread *See document structure, story*
- 27 thumbnail **10**
- 28 described 22, **32**
- 29 formats *See image*
- 30 parts *See parts*
- 31 relationship 25
- 32 usage 32
- 33 TIFF *See image, TIFF*
- 34 transformation
- 35 composability 270
- 36 described 186, **270**
- 37 effective transform *See transformation, composability*
- 38 elements applied to 171, 172
- 39 matrix
- 40 abbreviated syntax 187
- 41 abbreviated syntax example 190
- 42 described 186, 187
- 43 example 188
- 44 inverting x-axis 187
- 45 inverting y-axis 188
- 46 multiplying 187
- 47 positioning 188
- 48 rotating 188
- 49 scaling 187
- 50 skewing 188
- 51 specifying 186
- 52 mentioned 269
- 53 non-invertible transform, rendering of elements with ... 270
- 54 of brush 125
- 55 of canvas 61, 190
- 56 of geometry 77, 193
- 57 of glyphs 98, 192
- 58 of image brush 128, 195
- 59 of linear gradient brush 152, 199
- 60 of path 69, 191
- 61 of radial gradient brush 158, 200
- 62 of tiles 196
- 63 of tiles, example 198
- 64 of visual brush 131, 196
- 65 transparency *See opacity*
- 66 TrueType collection (TTC) font ... *See font:TrueType collection*
- 67 (TTC)
- 68 TrueType font *See font:TrueType*
-
- 69 **V**
- 70 vector graphics *See path*
- 71 versioning *See markup compatibility*
-
- 72 **W**
- 73 whitespace *See XPS Document format, XML, whitespace*
- 74 Windows Color System (WCS) *See color, color profile,*
- 75 Windows Color System (WCS) profile
- 76 Windows Media Photo *See image, Windows Media Photo*
-
- 77 **X**
- 78 XML *See XPS Document format, XML*
- 79 XML namespaces *See namespace; XPS Document format,*
- 80 XML, namespaces
- 81 XML Paper Specification
- 82 document format *See XPS Document format*
- 83 organization of 17
- 84 XPS Document format **10, 17, 21**
- 85 content types 402
- 86 described 1, 49
- 87 example 23
- 88 extensibility *See markup compatibility*
- 89 fixed payload 7, 21
- 90 fixed payload root 7, 21
- 91 illustrated 19
- 92 language *See language*
- 93 markup elements *See XPS elements*
- 94 parts 19, 21
- 95 payload 21
- 96 properties
- 97 attribute syntax 46
- 98 composability 45, 171
- 99 described 45
- 100 element syntax 46
- 101 model 45
- 102 ordering 46
- 103 property attribute 9
- 104 property element 9
- 105 property value 9, 45
- 106 property 9

1	relationship types.....	403	56	<FixedDocumentSequence>.....	49, 308
2	relationships.....	19, 24	57	<LinkTarget>.....	52, 321
3	root	49	58	<PageContent.LinkTargets>	51, 325
4	versioning.....	See markup compatibility	59	<PageContent>.....	50, 324
5	XML		60	page-level	
6	DTDs prohibited.....	44	61	<ArcSegment>	81, 301
7	markup design	43	62	<Canvas.Clip>	183, 304
8	markup model	45–47	63	<Canvas.OpacityMask>	203, 304
9	namespaces	45, 401	64	<Canvas.RenderTransform>.....	190, 305
10	Unicode encodings permitted	44	65	<Canvas.Resources>.....	175, 305
11	usage.....	44	66	<Canvas>	61, 302
12	whitespace.....	47	67	<FixedPage.Resources>.....	173, 310
13	XML and XSI namespace usage.....	45	68	<FixedPage>	53, 308
14	XML schema (XSD)		69	<Glyphs.Clip>.....	185, 315
15	characteristics.....	45	70	<Glyphs.Fill>	124, 315
16	discard control schema.....	387	71	<Glyphs.OpacityMask>.....	205, 315
17	document structure schema.....	381	72	<Glyphs.RenderTransform>	192, 316
18	resource dictionary key schema	379	73	<Glyphs>.....	98, 310
19	signature definitions schema.....	353	74	<GradientStop>	166, 316
20	validity requirement	44	75	<ImageBrush.Transform>.....	195, 318
21	XPS Document schema	355	76	<ImageBrush>	127, 317
22	XPS elements		77	<LinearGradientBrush.GradientStops>	157, 321
23	digital signature		78	<LinearGradientBrush.Transform>.....	199, 321
24	<Intent>	267, 319	79	<LinearGradientBrush>	152, 319
25	<SignatureDefinition>.....	265, 341	80	<MatrixTransform>	186, 323
26	<SignatureDefinitions>	265, 342	81	<Path.Clip>	184, 331
27	<SignBy>	268, 341	82	<Path.Data>.....	72, 331
28	<SigningLocation>.....	268, 342	83	<Path.Fill>.....	73, 331
29	<SpotLocation>	266, 343	84	<Path.OpacityMask>	204, 331
30	discard control		85	<Path.RenderTransform>	191, 332
31	<Discard>	259, 305	86	<Path.Stroke>.....	74, 332
32	<DiscardControl>	259, 306	87	<Path>	68, 326
33	document structure		88	<PathFigure>	80, 333
34	<DocumentOutline>	229, 306	89	<PathGeometry.Transform>	193, 334
35	<DocumentStructure.Outline>	229, 307	90	<PathGeometry>	77, 333
36	<DocumentStructure>	228, 307	91	<PolyBezierSegment>.....	85, 335
37	<FigureStructure>	246, 308	92	<PolyLineSegment>.....	86, 335
38	<ListItemStructure>	246, 322	93	<PolyQuadraticBezierSegment>.....	87, 336
39	<ListStructure>	245, 322	94	<RadialGradientBrush.GradientStops>	165, 339
40	<NamedElement>.....	247, 323	95	<RadialGradientBrush.Transform>.....	200, 339
41	<OutlineEntry>.....	229, 324	96	<RadialGradientBrush>	158, 336
42	<ParagraphStructure>	243, 325	97	<ResourceDictionary>	176, 339
43	<SectionStructure>	243, 340	98	<SolidColorBrush>	126, 342
44	<Story>.....	231, 343	99	<VisualBrush.Transform>	196, 350
45	<StoryBreak>	243, 344	100	<VisualBrush.Visual>	132, 351
46	<StoryFragment>	239, 344	101	<VisualBrush>.....	131, 348
47	<StoryFragmentReference>.....	232, 346			
48	<StoryFragments>.....	237, 345			
49	<TableCellStructure>	245, 346	102	Z	
50	<TableRowGroupStructure>	244, 347	103	ZIP	
51	<TableRowStructure>	244, 347	104	archive	10, 19
52	<TableStructure>	244, 348	105	item	10
53	document-level		106	utilities	35
54	<DocumentReference>	49, 306			
55	<FixedDocument>.....	50, 308			