

1

# Open XML

2

# Paper

3

# Specification

---

4

5 OpenXPS Specification and Reference Guide

6

7

8

9 Working Draft 1.5 (Interim Release 3)

10

11 February 2009

12

13

14 © 2007–2009 Ecma International. All rights reserved.

15

16

17

18 This document is "a work in progress" and was produced by Ecma Technical Committee TC46.



1	<b>Contents</b>	
2	<b>1. SCOPE</b> .....	<b>1</b>
3	<b>2. CONFORMANCE</b> .....	<b>3</b>
4	<b>2.1 Requirements Terminology</b> .....	<b>3</b>
5	<b>2.2 Implementation Conformance</b> .....	<b>3</b>
6	<b>2.3 Instantiating Error Conditions</b> .....	<b>4</b>
7	<b>3. NORMATIVE REFERENCES</b> .....	<b>5</b>
8	<b>4. DEFINITIONS</b> .....	<b>9</b>
9	<b>5. NOTATIONAL CONVENTIONS</b> .....	<b>13</b>
10	<b>5.1 Document Conventions</b> .....	<b>13</b>
11	<b>5.2 Diagrams</b> .....	<b>13</b>
12	<b>6. ACRONYMS AND ABBREVIATIONS</b> .....	<b>15</b>
13	<b>7. GENERAL DESCRIPTION</b> .....	<b>17</b>
14	<b>8. OPENXPS DOCUMENT FORMAT</b> .....	<b>19</b>
15	<b>8.1 How This Standard Is Organized</b> .....	<b>19</b>
16	<b>8.2 Package</b> .....	<b>21</b>
17	<b>9. PARTS AND RELATIONSHIPS</b> .....	<b>23</b>
18	<b>9.1 Fixed Payload</b> .....	<b>23</b>
19	9.1.1 Fixed Payload Relationships.....	26
20	9.1.2 FixedDocumentSequence Part.....	27
21	9.1.3 FixedDocument Part.....	27
22	9.1.4 FixedPage Part.....	28
23	9.1.5 Image Parts.....	28
24	9.1.6 Thumbnail Parts .....	34
25	9.1.7 Font Parts .....	35
26	9.1.8 Remote Resource Dictionary Parts .....	39
27	9.1.9 PrintTicket Parts.....	40
28	9.1.10 SignatureDefinitions Part .....	40
29	9.1.11 DocumentStructure Part .....	40
30	9.1.12 StoryFragments Part .....	41
31	<b>9.2 Part Naming Recommendations</b> .....	<b>41</b>
32	<b>9.3 OpenXPS Document Markup</b> .....	<b>43</b>
33	9.3.1 Support for Versioning and Extensibility .....	44
34	9.3.2 XML Usage .....	44
35	9.3.3 Markup Model .....	45
36	9.3.4 Whitespace.....	47
37	9.3.5 Language .....	47
38	<b>10. DOCUMENTS</b> .....	<b>49</b>
39	<b>10.1 &lt;FixedDocumentSequence&gt; Element</b> .....	<b>49</b>
40	10.1.1 <DocumentReference> Element.....	49
41	<b>10.2 &lt;FixedDocument&gt; Element</b> .....	<b>50</b>
42	10.2.1 <PageContent> Element.....	50
43	10.2.2 <PageContent.LinkTargets> Element .....	51
44	10.2.3 <LinkTarget> Element.....	52

1	<b>10.3</b>	<b>&lt;FixedPage&gt; Element</b> .....	<b>53</b>
2	10.3.1	BleedBox Attribute.....	54
3	10.3.2	ContentBox Attribute .....	55
4	10.3.3	Media Orientation and Scaling .....	55
5	<b>10.4</b>	<b>&lt;Canvas&gt; Element</b> .....	<b>56</b>
6	<b>10.5</b>	<b>&lt;Path&gt; Element</b> .....	<b>59</b>
7	<b>10.6</b>	<b>&lt;Glyphs&gt; Element</b> .....	<b>60</b>
8	<b>11. GRAPHICS</b> .....		<b>61</b>
9	<b>11.1</b>	<b>&lt;Path&gt; Element</b> .....	<b>62</b>
10	11.1.1	<Path.Data> Element .....	66
11	11.1.2	<Path.Fill> Element .....	67
12	11.1.3	<Path.Stroke> Element .....	68
13	<b>11.2</b>	<b>Geometries and Figures</b> .....	<b>69</b>
14	11.2.1	Geometries .....	71
15	11.2.2	Figures .....	74
16	11.2.3	Abbreviated Geometry Syntax .....	83
17	<b>12. TEXT</b> .....		<b>91</b>
18	<b>12.1</b>	<b>&lt;Glyphs&gt; Element</b> .....	<b>92</b>
19	12.1.1	Glyph Metrics.....	97
20	12.1.2	Mapping Code Units to Glyphs .....	98
21	12.1.3	Indices Attribute .....	102
22	12.1.4	UnicodeString Attribute .....	104
23	12.1.5	StyleSimulations Attribute .....	105
24	12.1.6	IsSideways Attribute .....	105
25	12.1.7	DeviceFontName Attribute .....	111
26	12.1.8	xml:lang Attribute .....	111
27	12.1.9	CaretStops Attribute .....	111
28	12.1.10	Optimizing Glyph Markup .....	112
29	12.1.11	Glyph Markup Examples .....	112
30	<b>12.2</b>	<b>&lt;Glyphs.Fill&gt; Element</b> .....	<b>115</b>
31	<b>13. BRUSHES</b> .....		<b>117</b>
32	<b>13.1</b>	<b>&lt;SolidColorBrush&gt; Element</b> .....	<b>118</b>
33	<b>13.2</b>	<b>&lt;ImageBrush&gt; Element</b> .....	<b>119</b>
34	<b>13.3</b>	<b>&lt;VisualBrush&gt; Element</b> .....	<b>123</b>
35	13.3.1	<VisualBrush.Visual> Element .....	124
36	<b>13.4</b>	<b>Common Attributes for Tiling Brushes</b> .....	<b>127</b>
37	13.4.1	Viewbox, Viewport, ViewboxUnits, and ViewportUnits Attributes .....	127
38	13.4.2	TileMode Attribute .....	132
39	<b>13.5</b>	<b>&lt;LinearGradientBrush&gt; Element</b> .....	<b>143</b>
40	13.5.1	SpreadMethod Attribute.....	145
41	13.5.2	<LinearGradientBrush.GradientStops> Element .....	148
42	<b>13.6</b>	<b>&lt;RadialGradientBrush&gt; Element</b> .....	<b>149</b>
43	13.6.1	SpreadMethod Attribute.....	152
44	13.6.2	<RadialGradientBrush.GradientStops> Element .....	155
45	<b>13.7</b>	<b>&lt;GradientStop&gt; Element</b> .....	<b>156</b>
46	<b>13.8</b>	<b>Using a Brush as an Opacity Mask</b> .....	<b>157</b>
47	<b>14. COMMON PROPERTIES</b> .....		<b>161</b>
48	<b>14.1</b>	<b>Opacity</b> .....	<b>162</b>
49	<b>14.2</b>	<b>Resources and Resource References</b> .....	<b>162</b>
50	14.2.1	<FixedPage.Resources> Element .....	163

1	14.2.2	<Canvas.Resources> Element .....	164
2	14.2.3	<ResourceDictionary> Element .....	165
3	14.2.4	Resource References .....	168
4	14.2.5	Scoping Rules for Resolving Resource References.....	169
5	14.2.6	Support for Markup Compatibility .....	170
6	<b>14.3</b>	<b>Clipping.....</b>	<b>170</b>
7	14.3.1	<Canvas.Clip> Element .....	170
8	14.3.2	<Path.Clip> Element .....	172
9	14.3.3	<Glyphs.Clip> Element.....	173
10	<b>14.4</b>	<b>Positioning Content .....</b>	<b>174</b>
11	14.4.1	<MatrixTransform> Element.....	174
12	14.4.2	<Canvas.RenderTransform> Element.....	178
13	14.4.3	<Path.RenderTransform> Element.....	179
14	14.4.4	<Glyphs.RenderTransform> Element .....	180
15	14.4.5	<PathGeometry.Transform> Element .....	181
16	14.4.6	<ImageBrush.Transform> Element .....	182
17	14.4.7	<VisualBrush.Transform> Element .....	183
18	14.4.8	<LinearGradientBrush.Transform> Element .....	186
19	14.4.9	<RadialGradientBrush.Transform> Element .....	187
20	<b>14.5</b>	<b>OpacityMask.....</b>	<b>190</b>
21	14.5.1	<Canvas.OpacityMask> Element .....	190
22	14.5.2	<Path.OpacityMask> Element .....	191
23	14.5.3	<Glyphs.OpacityMask> Element.....	192
24	<b>15</b>	<b>COLOR.....</b>	<b>195</b>
25	<b>15.1</b>	<b>Color Support .....</b>	<b>195</b>
26	15.1.1	sRGB Color Space .....	196
27	15.1.2	scRGB Color Space.....	196
28	15.1.3	Gray Color Space.....	196
29	15.1.4	CMYK Color Space.....	196
30	15.1.5	N-Channel Color Spaces.....	196
31	15.1.6	Named Color for Spot Colors and N-tone Images.....	196
32	15.1.7	Identifying Output-Ready Color Spaces Using ICC Profiles .....	196
33	15.1.8	ICC Profiles .....	196
34	<b>15.2</b>	<b>Vector Color Syntax .....</b>	<b>197</b>
35	15.2.1	sRGB Color Syntax.....	199
36	15.2.2	scRGB Color Syntax .....	199
37	15.2.3	Grayscale syntax .....	200
38	15.2.4	CMYK Color Syntax .....	200
39	15.2.5	N-Channel Color Syntax.....	201
40	15.2.6	Named Color Syntax .....	202
41	<b>15.3</b>	<b>Colors in Raster Images.....</b>	<b>203</b>
42	15.3.1	sRGB Raster Images .....	203
43	15.3.2	scRGB Raster Images .....	204
44	15.3.3	Gray Raster Images .....	204
45	15.3.4	CMYK Raster Images .....	205
46	15.3.5	N-channel Raster Images .....	205
47	15.3.6	Named Color Raster Images .....	206
48	15.3.7	Device Color Raster Images.....	207
49	15.3.8	Images and Color Profile Association.....	207
50	<b>15.4</b>	<b>Registration Marks for Color Separations.....</b>	<b>209</b>
51	<b>15.5</b>	<b>Alpha and Gradient Blending .....</b>	<b>209</b>
52	<b>15.6</b>	<b>Color Rendering Intent .....</b>	<b>210</b>
53	<b>16</b>	<b>DOCUMENT STRUCTURE AND INTERACTIVITY.....</b>	<b>211</b>
54	<b>16.1</b>	<b>Document Structure Markup.....</b>	<b>211</b>
55	16.1.1	DocumentStructure Part .....	211
56	16.1.2	StoryFragments Part .....	217
57	<b>16.2</b>	<b>Hyperlinks.....</b>	<b>231</b>

1 16.2.1 Hyperlink Activation .....231

2 16.2.2 Hyperlink Addressing .....232

3 16.2.3 Name Attribute .....232

4 16.2.4 FixedPage.NavigateUri Attribute.....233

5 **16.3 Selection ..... 234**

6 **16.4 Accessibility ..... 234**

7 16.4.1 Reading Order .....234

8 16.4.2 Screen Reader Applications .....235

9 16.4.3 Text Alternatives for Graphics and Images.....235

10 **17. OPENXPS DOCUMENT PACKAGE FEATURES..... 237**

11 **17.1 Interleaving Optimizations ..... 237**

12 17.1.1 Empty PrintTicket .....239

13 17.1.2 Optimizing Interleaving Order.....239

14 17.1.3 Consuming Interleaved Packages .....243

15 17.1.4 Consumers with Resource Constraints.....243

16 17.1.5 Interleaving Optimizations and Digital Signatures .....245

17 **17.2 Digital Signatures ..... 246**

18 17.2.1 Signature Policy .....246

19 17.2.2 Signature Definitions.....249

20 **17.3 Core Properties ..... 253**

21 **18. RENDERING RULES ..... 255**

22 **18.1 Coordinate System and Rendering Placement ..... 255**

23 18.1.1 Page Dimensions .....255

24 18.1.2 Rounding of Coordinates.....255

25 18.1.3 Transforms.....256

26 18.1.4 Pixel Center Location, Pixel Placement, and Pixel Inclusion .....256

27 18.1.5 Maximum Placement Error .....257

28 18.1.6 Pixel Placement for Glyphs .....257

29 18.1.7 Abutment of Shapes .....257

30 18.1.8 Clipping Behavior .....257

31 **18.2 Implementation Limits ..... 258**

32 **18.3 Gradient Computations ..... 259**

33 18.3.1 All Gradients.....259

34 18.3.2 Linear Gradients.....261

35 18.3.3 Radial Gradients.....263

36 **18.4 Opacity Computations..... 266**

37 18.4.1 Pre-Multiplied Alpha and Superluminous Colors .....268

38 **18.5 Composition Rules ..... 269**

39 18.5.1 Optimization Guidelines .....270

40 18.5.2 Composition Examples .....271

41 **18.6 Stroke Rendering..... 274**

42 18.6.1 Stroke Edge Parallelization .....274

43 18.6.2 Phase Control .....274

44 18.6.3 Symmetry of Stroke Drawing Algorithms .....274

45 18.6.4 Rules for Dash Cap Rendering.....275

46 18.6.5 Rules for Line Cap Rendering.....277

47 18.6.6 Line Caps for Dashed Strokes .....278

48 18.6.7 Rules for Line Join Rendering.....279

49 18.6.8 Rules for Degenerate Line and Curve Segments.....283

50 18.6.9 Stroking and Fill Rule .....284

51 18.6.10 Mixing Stroked and Non-Stroked Segments .....284

52 18.6.11 Stroke Behavior with Multiple Path Figures .....284

53 18.6.12 Consistent Nominal Stroke Width .....284

54 **18.7 Brushes and Images ..... 285**

55 18.7.1 Small Tiles .....285

56 18.7.2 Image Scaling.....285

1	18.7.3	Tile Placement.....	286
2	18.7.4	Tiling Transparent Visual Brushes and Image Brushes.....	286
3	<b>19.</b>	<b>ELEMENTS.....</b>	<b>287</b>
4	<b>19.1</b>	<b>ArcSegment.....</b>	<b>287</b>
5	<b>19.2</b>	<b>Canvas.....</b>	<b>288</b>
6	<b>19.3</b>	<b>Canvas.Clip.....</b>	<b>290</b>
7	<b>19.4</b>	<b>Canvas.OpacityMask.....</b>	<b>290</b>
8	<b>19.5</b>	<b>Canvas.RenderTransform.....</b>	<b>291</b>
9	<b>19.6</b>	<b>Canvas.Resources.....</b>	<b>291</b>
10	<b>19.7</b>	<b>Discard.....</b>	<b>291</b>
11	<b>19.8</b>	<b>DiscardControl.....</b>	<b>292</b>
12	<b>19.9</b>	<b>DocumentOutline.....</b>	<b>292</b>
13	<b>19.10</b>	<b>DocumentReference.....</b>	<b>292</b>
14	<b>19.11</b>	<b>DocumentStructure.....</b>	<b>293</b>
15	<b>19.12</b>	<b>DocumentStructure.Outline.....</b>	<b>293</b>
16	<b>19.13</b>	<b>FigureStructure.....</b>	<b>294</b>
17	<b>19.14</b>	<b>FixedDocument.....</b>	<b>294</b>
18	<b>19.15</b>	<b>FixedDocumentSequence.....</b>	<b>294</b>
19	<b>19.16</b>	<b>FixedPage.....</b>	<b>294</b>
20	<b>19.17</b>	<b>FixedPage.Resources.....</b>	<b>296</b>
21	<b>19.18</b>	<b>Glyphs.....</b>	<b>296</b>
22	<b>19.19</b>	<b>Glyphs.Clip.....</b>	<b>301</b>
23	<b>19.20</b>	<b>Glyphs.Fill.....</b>	<b>301</b>
24	<b>19.21</b>	<b>Glyphs.OpacityMask.....</b>	<b>301</b>
25	<b>19.22</b>	<b>Glyphs.RenderTransform.....</b>	<b>302</b>
26	<b>19.23</b>	<b>GradientStop.....</b>	<b>302</b>
27	<b>19.24</b>	<b>ImageBrush.....</b>	<b>303</b>
28	<b>19.25</b>	<b>ImageBrush.Transform.....</b>	<b>304</b>
29	<b>19.26</b>	<b>Intent.....</b>	<b>305</b>
30	<b>19.27</b>	<b>LinearGradientBrush.....</b>	<b>305</b>
31	<b>19.28</b>	<b>LinearGradientBrush.GradientStops.....</b>	<b>307</b>
32	<b>19.29</b>	<b>LinearGradientBrush.Transform.....</b>	<b>307</b>
33	<b>19.30</b>	<b>LinkTarget.....</b>	<b>308</b>
34	<b>19.31</b>	<b>ListItemStructure.....</b>	<b>308</b>
35	<b>19.32</b>	<b>ListStructure.....</b>	<b>309</b>
36	<b>19.33</b>	<b>MatrixTransform.....</b>	<b>309</b>
37	<b>19.34</b>	<b>NamedElement.....</b>	<b>309</b>
38	<b>19.35</b>	<b>OutlineEntry.....</b>	<b>310</b>
39	<b>19.36</b>	<b>PageContent.....</b>	<b>311</b>
40	<b>19.37</b>	<b>PageContent.LinkTargets.....</b>	<b>311</b>
41	<b>19.38</b>	<b>ParagraphStructure.....</b>	<b>312</b>

1	<b>19.39 Path</b> .....	<b>312</b>
2	<b>19.40 Path.Clip</b> .....	<b>317</b>
3	<b>19.41 Path.Data</b> .....	<b>317</b>
4	<b>19.42 Path.Fill</b> .....	<b>317</b>
5	<b>19.43 Path.OpacityMask</b> .....	<b>317</b>
6	<b>19.44 Path.RenderTransform</b> .....	<b>318</b>
7	<b>19.45 Path.Stroke</b> .....	<b>318</b>
8	<b>19.46 PathFigure</b> .....	<b>319</b>
9	<b>19.47 PathGeometry</b> .....	<b>319</b>
10	<b>19.48 PathGeometry.Transform</b> .....	<b>320</b>
11	<b>19.49 PolyBezierSegment</b> .....	<b>321</b>
12	<b>19.50 PolyLineSegment</b> .....	<b>321</b>
13	<b>19.51 PolyQuadraticBezierSegment</b> .....	<b>322</b>
14	<b>19.52 RadialGradientBrush</b> .....	<b>322</b>
15	<b>19.53 RadialGradientBrush.GradientStops</b> .....	<b>325</b>
16	<b>19.54 RadialGradientBrush.Transform</b> .....	<b>325</b>
17	<b>19.55 ResourceDictionary</b> .....	<b>325</b>
18	<b>19.56 SectionStructure</b> .....	<b>326</b>
19	<b>19.57 SignBy</b> .....	<b>327</b>
20	<b>19.58 SignatureDefinition</b> .....	<b>327</b>
21	<b>19.59 SignatureDefinitions</b> .....	<b>328</b>
22	<b>19.60 SigningLocation</b> .....	<b>328</b>
23	<b>19.61 SolidColorBrush</b> .....	<b>328</b>
24	<b>19.62 SpotLocation</b> .....	<b>329</b>
25	<b>19.63 Story</b> .....	<b>329</b>
26	<b>19.64 StoryBreak</b> .....	<b>330</b>
27	<b>19.65 StoryFragment</b> .....	<b>330</b>
28	<b>19.66 StoryFragments</b> .....	<b>331</b>
29	<b>19.67 StoryFragmentReference</b> .....	<b>332</b>
30	<b>19.68 TableCellStructure</b> .....	<b>332</b>
31	<b>19.69 TableRowGroupStructure</b> .....	<b>333</b>
32	<b>19.70 TableRowStructure</b> .....	<b>333</b>
33	<b>19.71 TableStructure</b> .....	<b>334</b>
34	<b>19.72 VisualBrush</b> .....	<b>334</b>
35	<b>19.73 VisualBrush.Transform</b> .....	<b>336</b>
36	<b>19.74 VisualBrush.Visual</b> .....	<b>337</b>
37	<b>A. SCHEMAS – W3C XML</b> .....	<b>339</b>
38	<b>A.1 Signature Definitions</b> .....	<b>339</b>
39	<b>A.2 OpenXPS Document</b> .....	<b>341</b>
40	<b>A.3 Resource Dictionary Key</b> .....	<b>365</b>
41	<b>A.4 Document Structure</b> .....	<b>367</b>



1	<b>A.5 Discard Control</b> .....	<b>373</b>
2	<b>A.6 3D-Graphic Content</b> .....	<b>374</b>
3	<b>B. SCHEMAS – RELAX NG</b> .....	<b>377</b>
4	<b>B.1 Signature Definitions</b> .....	<b>377</b>
5	<b>B.2 OpenXPS Document</b> .....	<b>379</b>
6	<b>B.3 Resource Dictionary Key</b> .....	<b>381</b>
7	<b>B.4 Document Structure</b> .....	<b>383</b>
8	<b>B.5 Discard Control</b> .....	<b>385</b>
9	<b>B.6 3D-Graphic Content</b> .....	<b>386</b>
10	<b>C. ABBREVIATED GEOMETRY SYNTAX ALGORITHM</b> .....	<b>387</b>
11	<b>D. STANDARD NAMESPACES AND CONTENT TYPES</b> .....	<b>393</b>
12	<b>D.1 XML Namespace URIs</b> .....	<b>393</b>
13	<b>D.2 Content Types</b> .....	<b>394</b>
14	<b>D.3 Relationship Types</b> .....	<b>395</b>
15	<b>E. RECOMMENDED FILE NAME EXTENSION AND CONTENT TYPES</b> .....	<b>397</b>
16	<b>E.1 Identification of OpenXPS Documents</b> .....	<b>397</b>
17	<b>E.2 Embedding Producer Identification</b> .....	<b>397</b>
18	<b>E.3 Determination of OPC payload</b> .....	<b>397</b>
19	<b>F. CONFORMANCE REQUIREMENTS</b> .....	<b>399</b>
20	<b>F.1 Implementation Conformance</b> .....	<b>399</b>
21	F.1.1 MUST Conformance Requirements .....	399
22	F.1.2 SHOULD Conformance Requirements .....	400
23	<b>F.2 OpenXPS Document Format</b> .....	<b>400</b>
24	F.2.1 MUST Conformance Requirements .....	400
25	F.2.2 SHOULD Conformance Requirements .....	400
26	<b>F.3 Parts and Relationships</b> .....	<b>400</b>
27	F.3.1 MUST Conformance Requirements .....	400
28	F.3.2 SHOULD Conformance Requirements .....	408
29	F.3.3 OPTIONAL Conformance Requirements .....	412
30	<b>F.4 Documents</b> .....	<b>414</b>
31	F.4.1 MUST Conformance Requirements .....	414
32	F.4.2 SHOULD Conformance Requirements .....	416
33	F.4.3 OPTIONAL Conformance Requirements .....	416
34	<b>F.5 Graphics</b> .....	<b>417</b>
35	F.5.1 MUST Conformance Requirements .....	417
36	F.5.2 SHOULD Conformance Requirements .....	417
37	F.5.3 OPTIONAL Conformance Requirements .....	418
38	<b>F.6 Text</b> .....	<b>418</b>
39	F.6.1 MUST Conformance Requirements .....	418
40	F.6.2 SHOULD Conformance Requirements .....	420
41	F.6.3 OPTIONAL Conformance Requirements .....	421
42	<b>F.7 Brushes</b> .....	<b>422</b>
43	F.7.1 MUST Conformance Requirements .....	422
44	<b>F.8 Common Properties</b> .....	<b>422</b>
45	F.8.1 MUST Conformance Requirements .....	422
46	F.8.2 SHOULD Conformance Requirements .....	423

1 F.8.3 OPTIONAL Conformance Requirements.....424

2 **F.9 Color..... 424**

3 F.9.1 MUST Conformance Requirements .....424

4 F.9.2 SHOULD Conformance Requirements .....428

5 F.9.3 OPTIONAL Conformance Requirements.....431

6 **F.10 Document Structure and Interactivity ..... 433**

7 F.10.1MUST Conformance Requirements .....433

8 F.10.2SHOULD Conformance Requirements .....434

9 F.10.3OPTIONAL Conformance Requirements.....436

10 **F.11 OpenXPS Document Package Features..... 437**

11 F.11.1MUST Conformance Requirements .....437

12 F.11.2SHOULD Conformance Requirements .....439

13 F.11.3OPTIONAL Conformance Requirements.....441

14 **F.12 Rendering Rules ..... 442**

15 F.12.1MUST Conformance Requirements .....442

16 F.12.2SHOULD Conformance Requirements .....443

17 F.12.3OPTIONAL Conformance Requirements.....447

18 **F.13 Additional Conformance Requirements ..... 448**

19 F.13.1MUST Conformance Requirements .....448

20 **F.14 3D Graphic Content ..... 449**

21 F.14.1MUST Conformance Requirements .....449

22 F.14.2SHOULD Conformance Requirements .....450

23 F.14.3OPTIONAL Conformance Requirements.....451

24 **F.15 Recommended File Name Extension and Content Types ..... 451**

25 F.15.1MUST Conformance Requirements .....451

26 F.15.2SHOULD Conformance Requirements .....451

27 **G. 3D GRAPHIC CONTENT..... 453**

28 **G.1 Brush3D..... 457**

29 **H. BIBLIOGRAPHY ..... 463**

30 **I. INDEX..... 465**

31

## 1 List of Figures

2	Figure 8–1. Package-based OpenXPS Document format .....	21
3	Figure 11–1. Fill using EvenOdd algorithm.....	73
4	Figure 11–2. Fill using NonZero algorithm .....	73
5	Figure 11–3. Arc choice A.....	77
6	Figure 11–4. Arc choice B.....	77
7	Figure 11–5. Arc choice C.....	77
8	Figure 11–6. Arc choice D .....	77
9	Figure 12–1. Glyph metrics .....	97
10	Figure 12–2. Upright (usually horizontal) glyph metrics.....	97
11	Figure 12–3. Sideways (usually vertical) glyph metrics .....	98
12	Figure 17–1. A sample signature spot .....	252
13	Figure 18–1. Extreme curvatures and dash rendering .....	274
14	Figure 18–2. Flat dash caps.....	275
15	Figure 18–3. Square dash caps .....	276
16	Figure 18–4. Round dash caps .....	276
17	Figure 18–5. Triangular dash caps.....	277
18	Figure 18–6. Overlapping dash segments .....	277
19	Figure 18–7. Flat start line cap, flat end line cap .....	278
20	Figure 18–8. Square start line cap, square end line cap .....	278
21	Figure 18–9. Triangular start line cap, triangular end line cap .....	278
22	Figure 18–10. Round start line cap, round end line cap .....	278
23	Figure 18–11. Stroke start or end point within a dash for flat dash caps.....	278
24	Figure 18–12. Stroke start or end point within a dash for non-flat dash caps .....	278
25	Figure 18–13. Stroke start or end point within a gap for flat dash caps .....	279
26	Figure 18–14. Stroke start or end point within a gap for not-flat dash caps .....	279
27	Figure 18–15. Round line join with right angle .....	279
28	Figure 18–16. Round line join with acute angle .....	280
29	Figure 18–17. Round line join with obtuse angle .....	280
30	Figure 18–18. Beveled line join with right angle .....	280
31	Figure 18–19. Beveled line join with acute angle .....	281
32	Figure 18–20. Beveled line join with obtuse angle .....	281
33	Figure 18–21. Mitered line join with right angle and miter limit of 1.0 .....	282
34	Figure 18–22. Mitered line join with acute angle and miter limit of 1.0 .....	282
35	Figure 18–23. Mitered line join with obtuse angle and miter limit of 1.0 .....	282
36	Figure 18–24. Mitered line join with right angle and miter limit of 2.0 .....	283
37	Figure 18–25. Mitered line join with acute angle and miter limit of 2.0 .....	283
38	Figure 18–26. Mitered line join with acute angle and miter limit of 10.0 .....	283

39



## 1 List of Tables

2	Table 9–1. OpenXPS Document parts.....	23
3	Table 9–2. Fixed payload relationships.....	26
4	Table 9–3. Supported JPEG APPn markers.....	29
5	Table 9–4. Support for ancillary PNG chunks.....	30
6	Table 9–5. Supported TIFF tags.....	30
7	Table 9–6. Supported JPEG XR features.....	33
8	Table 9–7. Guidelines for Open Font Format embedding.....	36
9	Table 9–8. Cmap table selection.....	39
10	Table 11–1. Arc segment definition.....	76
11	Table 11–2. Commands.....	84
12	Table 12–3. Glyph specifications.....	102
13	Table 12–4. Portions of the cluster specification.....	103
14	Table 12–5. IsSideways and BidiLevel effects on origin placement.....	107
15	Table 13–1. Brush types.....	117
16	Table 13–2. Common attributes for <ImageBrush> and <VisualBrush> elements.....	127
17	Table 14–1. Common property attributes.....	161
18	Table 14–2. Common property elements.....	162
19	Table 15–1. Syntax summary.....	198
20	Table 15–2. Color Space Pixel Format Defaults.....	208
21	Table 15–3. Recommended ICC rendering intent usage.....	210
22	Table 16–1. StoryFragments part elements.....	218
23	Table 16–2. Unicode character categories.....	233
24	Table 18–1. Recommended minimum processing requirements.....	258
25	Table 18–2. Opacity computation symbols.....	266
26	Table D–1. Package-wide namespaces.....	393
27	Table D–2. OpenXPS Document namespaces.....	393
28	Table D–3. Package-wide content types.....	394
29	Table D–4. OpenXPS Document content types.....	394
30	Table D–5. Package-wide relationship types.....	395
31	Table D–6. OpenXPS Document relationship types.....	395
32	Table F–1. Implementation MUST conformance requirements.....	399
33	Table F–2. Implementation SHOULD conformance requirements.....	400
34	Table F–3. OpenXPS Document format MUST conformance requirements.....	400
35	Table F–4. OpenXPS Document format SHOULD conformance requirements.....	400
36	Table F–5. Parts and Relationships MUST conformance requirements.....	400
37	Table F–6. Parts and Relationships SHOULD conformance requirements.....	408
38	Table F–7. Parts and Relationships OPTIONAL conformance requirements.....	412
39	Table F–8. Document MUST conformance requirements.....	414
40	Table F–9. Document SHOULD conformance requirements.....	416
41	Table F–10. Document OPTIONAL conformance requirements.....	416
42	Table F–11. Graphics MUST conformance requirements.....	417
43	Table F–12. Graphics SHOULD conformance requirements.....	417
44	Table F–13. Graphics OPTIONAL conformance requirements.....	418
45	Table F–14. Text MUST conformance requirements.....	418
46	Table F–15. Text SHOULD conformance requirements.....	420
47	Table F–16. Text OPTIONAL conformance requirements.....	421
48	Table F–17. Brushes MUST conformance requirements.....	422
49	Table F–18. Common properties MUST conformance requirements.....	422
50	Table F–19. Common properties SHOULD conformance requirements.....	423
51	Table F–20. Common properties OPTIONAL conformance requirements.....	424
52	Table F–21. Color MUST conformance requirements.....	424

1 Table F-22. Color SHOULD conformance requirements ..... 428  
2 Table F-23. Color OPTIONAL conformance requirements ..... 431  
3 Table F-24. Document structure MUST conformance requirements ..... 433  
4 Table F-25. Document structure SHOULD conformance requirements ..... 434  
5 Table F-26. Document structure OPTIONAL conformance requirements ..... 436  
6 Table F-27. OpenXPS Document package feature MUST conformance requirements ..... 437  
7 Table F-28. OpenXPS Document package feature SHOULD conformance requirements ..... 439  
8 Table F-29. OpenXPS Document package feature OPTIONAL conformance requirements ..... 441  
9 Table F-30. Rendering rules MUST conformance requirements ..... 442  
10 Table F-31. Rendering rules SHOULD conformance requirements ..... 443  
11 Table F-32. Rendering rules OPTIONAL conformance requirements ..... 447  
12 Table F-33. Additional MUST conformance requirements ..... 448  
13 Table F-34. 3D Graphic Content MUST conformance requirements ..... 449  
14 Table F-35. 3D Graphic Content SHOULD conformance requirements ..... 450  
15 Table F-36. 3D Graphic Content OPTIONAL conformance requirements ..... 451  
16 Table F-37. Recommended File Name Extension and Content Types MUST conformance  
17 requirements ..... 451  
18 Table F-38. Recommended File Name Extension and Content Types SHOULD conformance  
19 requirements ..... 451

20

## 1 List of Examples

2	Example 9–1. A typical OpenXPS Document .....	25
3	Example 9–2. OpenXPS Document part naming .....	43
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.....	58
12	Example 11–1. <Path.Data> usage .....	66
13	Example 11–2. <Path.Fill> usage.....	68
14	Example 11–3. <Path.Stroke> usage .....	69
15	Example 11–4. <PathGeometry> usage.....	72
16	Example 11–5. <ArcSegment> usage.....	77
17	Example 11–6. <PolyBezierSegment> usage .....	79
18	Example 11–7. <PolyLineSegment> usage.....	80
19	Example 11–8. <PolyQuadraticBezierSegment> usage .....	82
20	Example 11–9. Closed <PathFigure> usage.....	82
21	Example 11–10. A path described using abbreviated syntax .....	86
22	Example 11–11. Smooth Bézier curve.....	87
23	Example 11–12. Relative commands and curves .....	88
24	Example 12–1. One-to-one cluster map .....	99
25	Example 12–2. Many-to-one cluster map .....	99
26	Example 12–3. One-to-many cluster map .....	100
27	Example 12–4. Many-to-many cluster map.....	101
28	Example 12–5. Using indices to specify advance width.....	103
29	Example 12–6. Using the Indices attribute to specify glyph replacement for a cluster .....	104
30	Example 12–7. Text with positive uOffset and vOffset Indices values.....	107
31	Example 12–8. Right-to-left text (odd BidiLevel) .....	108
32	Example 12–9. Sideways text (IsSideways set to true) .....	108
33	Example 12–10. Vertical text.....	109
34	Example 12–11. Japanese vertical text .....	109
35	Example 12–12. Using the CaretStops attribute to determine a valid caret stop position .....	112
36	Example 12–13. Basic italic font.....	112
37	Example 12–14. Italic font using StyleSimulations attribute.....	113
38	Example 12–15. Kerning .....	113
39	Example 12–16. Ligatures .....	114
40	Example 12–17. Cluster maps .....	114
41	Example 13–1. <SolidColorBrush> usage.....	119
42	Example 13–2. <ImageBrush> usage.....	121
43	Example 13–3. <VisualBrush.Visual> usage .....	125
44	Example 13–4. ViewboxUnits and ViewportUnits attribute usage .....	128
45	Example 13–5. Tiling brush base image and rendering.....	128
46	Example 13–6. Tiling brush Viewport adjustments.....	129
47	Example 13–7. Tiling brush viewbox adjustments.....	130
48	Example 13–8. Image brush with a Viewbox larger than the image .....	131
49	Example 13–9. Image brush with TileMode value of None .....	132
50	Example 13–10. Visual brush with TileMode value of None .....	134
51	Example 13–11. Image brush with a TileMode value of Tile .....	135
52	Example 13–12. Visual brush with a TileMode value of Tile.....	136

1	Example 13–13. Image brush with a TileMode value of FlipX .....	137
2	Example 13–14. Visual brush with a TileMode value of FlipX .....	138
3	Example 13–15. Image brush with a TileMode value of FlipY.....	139
4	Example 13–16. Visual Brush with a TileMode value of FlipY .....	140
5	Example 13–17. Image brush with a TileMode value of FlipXY.....	141
6	Example 13–18. Visual brush with a TileMode value of FlipXY .....	142
7	Example 13–19. <LinearGradientBrush> usage .....	144
8	Example 13–20. Linear gradient brush with a SpreadMethod value of Pad .....	145
9	Example 13–21. Linear gradient brush with a SpreadMethod value of Reflect.....	146
10	Example 13–22. Linear gradient brush with a SpreadMethod value of Repeat .....	147
11	Example 13–23. A radial gradient brush.....	151
12	Example 13–24. RadialGradientBrush usage .....	151
13	Example 13–25. Radial gradient brush with a SpreadMethod value of Pad .....	152
14	Example 13–26. Radial gradient brush with a SpreadMethod value of Reflect.....	153
15	Example 13–27. Radial gradient brush with a SpreadMethod value of Repeat .....	154
16	Example 13–28. Opacity mask with linear gradient.....	157
17	Example 13–29. Opacity mask with radial gradient.....	158
18	Example 14–1. <FixedPage.Resources> usage .....	163
19	Example 14–2. <Canvas.Resources> usage .....	164
20	Example 14–3. Resource dictionary markup .....	166
21	Example 14–4. A remote resource dictionary and reference.....	167
22	Example 14–5. Using a resource reference to fill a brush.....	168
23	Example 14–6. Using scoping rules .....	169
24	Example 14–7. Canvas clip markup and rendering.....	171
25	Example 14–8. <Path.Clip> usage .....	172
26	Example 14–9. <Glyphs.Clip> usage .....	173
27	Example 14–10. Matrix scaling.....	175
28	Example 14–11. Matrix reversing the x axis .....	175
29	Example 14–12. Matrix reversing the y axis .....	175
30	Example 14–13. Matrix skewing.....	175
31	Example 14–14. Matrix Rotating .....	176
32	Example 14–15. Matrix positioning .....	176
33	Example 14–16. <MatrixTransform> usage .....	176
34	Example 14–17. Using abbreviated matrix transformation syntax.....	178
35	Example 14–18. <Canvas.RenderTransform> usage.....	178
36	Example 14–19. <Path.RenderTransform> usage.....	179
37	Example 14–20. <Glyphs.RenderTransform> usage .....	180
38	Example 14–21. <PathGeometry.Transform> usage.....	181
39	Example 14–22. <ImageBrush.Transform> usage.....	182
40	Example 14–23. <VisualBrush.Transform> usage .....	183
41	Example 14–24. <VisualBrush.Transform> usage with tiling behavior.....	185
42	Example 14–25. <LinearGradientBrush.Transform> usage .....	186
43	Example 14–26. <RadialGradientBrush.Transform> usage .....	188
44	Example 14–27. <Canvas.OpacityMask> usage .....	190
45	Example 14–28. <Path.OpacityMask> usage .....	191
46	Example 14–29. <Glyphs.OpacityMask> usage .....	193
47	Example 16–1. Document structure markup .....	212
48	Example 16–2. Document outline markup .....	214
49	Example 16–3. Simple multi-story document.....	217
50	Example 16–4. Story flowing back and forth across a page boundary .....	217
51	Example 16–5. Content structure spanning pages .....	219
52	Example 16–6. StoryFragments part markup.....	224
53	Example 16–7. Story fragments markup using a fragment name.....	225



1	Example 16–8. A relative, internal, named-address hyperlink .....	232
2	Example 16–9. A relative internal page address hyperlink .....	232
3	Example 17–1. Optimized interleaving for a single-threaded parsing architecture .....	239
4	Example 17–2. Optimized interleaving for a multi-threaded parsing architecture .....	242
5	Example 17–4. A SignatureDefinitions part .....	249
6	Example 18–1. Path opacity behavior for overlapping path figures .....	271
7	Example 18–2. Opacity behavior of path stroke intersections .....	271
8	Example 18–3. Opacity behavior of paths with stroked edges .....	272
9	Example G–1. 3D graphics content in FixedPage.fpage .....	454
10	Example G–2. 3D graphics content in FixedPage.fpage .....	456

11



## 1. Scope

2 This Standard defines *OpenXPS*, the [Open](#) XML Paper Specification. OpenXPS describes a 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 OpenXPS content.

6 A primary goal is to ensure the interoperability of independently created software and hardware  
7 systems that produce or consume OpenXPS content. This Standard defines the requirements  
8 that systems processing OpenXPS Documents must satisfy in order to achieve interoperability.

9 This Standard describes a paginated-document format called the *OpenXPS Document*. The  
10 format requirements are an extension of the packaging requirements described in the Open  
11 Packaging Conventions (OPC) Standard. That Standard 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 OpenXPS Standard, and are included by reference.

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



## 2. Conformance

---

### 2.1 Requirements Terminology

In this Standard, 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 Standard.
- **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**.

Requirements are documented inline in this Standard, and each requirement is denoted by a letter (M – **MUST**; S – **SHOULD**; O – **OPTIONAL**) and a unique 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 §Annex KKKKKKKKKKKKKKKKKKKKK.

---

### 2.2 Implementation Conformance

This Standard includes the implementation requirements that systems processing OpenXPS content must satisfy in order to achieve conforming interoperability. An implementation is a consumer, or a producer, or both a consumer and a producer.

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

- It **MUST** interpret and process the contents of OpenXPS Document instances in a manner conforming to this Standard [M0.1]. A consumer is **NOT REQUIRED** to interpret or process all of the content in an OpenXPS Document instance [M0.2].
- It **SHOULD** instantiate an error condition when OpenXPS Document content not conforming to this Standard is encountered [S0.1].
- It **MUST NOT** instantiate an error condition in response to OpenXPS Document content conforming to this Standard [M0.3].
- When "OPTIONAL" or "RECOMMENDED" features contained within OpenXPS Document instances are accessed by a consumer, the consumer **MUST** interpret and process those features in a manner conforming to this Standard [M0.4].

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

- Any OpenXPS Document instances it creates **MUST** conform to this Standard [M0.5].

- 1 • It MUST NOT introduce any non-conforming OpenXPS Document content when modifying  
2 an OpenXPS Document instance [M0.6].
  - 3 • When a producer chooses to use an "OPTIONAL" or "RECOMMENDED" feature in an  
4 OpenXPS Document instance, then the producer MUST create or modify that feature in a  
5 manner conforming to this Standard [M0.7].
- 

### 6 **2.3 Instantiating Error Conditions**

7 OpenXPS Documents are intended to address the requirements of a wide range of scenarios.  
8 The methods and effects of instantiated error conditions in response to conformance rule  
9 violations are implementation-defined.

10 [*Note*: Implementors are encouraged to instantiate error conditions to indicate non-conformant  
11 OpenXPS Documents where users can be expected to be able to act on the error information.  
12 Implementors are strongly encouraged to fail gracefully when processing non-compliant  
13 OpenXPS Documents to ensure that non-compliant OpenXPS Document instances, and non-  
14 compliant OpenXPS producers, do not proliferate. *end note*]

### 3. Normative References

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this Standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this Standard 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](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.jeta.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](http://www.color.org/ICC_Minor_Revision_for_Web.pdf)

IEC 61966:1999, *Multimedia systems and equipment - Colour measurement and management - Part 2-1: Colour management - Default RGB colour space - sRGB*

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 [ISO/IEC 14496-22:2007 Information technology — Coding of audio-visual objects — Part 22:](#)  
2 [Open Font Format](#)
- 3 ISO/IEC FCD 19775-1r1:200x *Information technology — Computer graphics and image*  
4 *processing — Extensible 3D (X3D) — Part 1: Architecture and base components.*
- 5 ISO/IEC 19776-1:2005 *Information technology — Computer graphics and image processing —*  
6 *Extensible 3D (X3D) encodings — Part 1: XML encoding.*
- 7 ISO/IEC 19776-2:2005 *Information technology — Computer graphics and image processing —*  
8 *Extensible 3D (X3D) encodings — Part 2: Classic VRML encoding.*
- 9 ISO/IEC 19776-3:2007 *Information technology — Computer graphics, image processing and*  
10 *environmental representation — Extensible (X3D) encodings — Part 3: Compressed binary*  
11 *encoding.*
- 12 [ISO/IEC 29199-2, JPEG XR Image Coding Specification](#)<sup>[I2281]</sup>
- 13 *JPEG File Interchange Format, Version 1.02.* Hamilton, Eric. World Wide Web Consortium. 1992.  
14 <http://www.w3.org/Graphics/JPEG/jfif3.pdf>
- 15 *Namespaces in XML 1.0 (Second Edition).* Bray, Tim, Dave Hollander, Andrew Layman, and  
16 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/)  
17 [xml-names-20060816/](http://www.w3.org/TR/2006/REC-xml-names-20060816/)
- 18 *Portable Network Graphics (PNG) Specification.* Duce, David (editor). Second Edition. World  
19 Wide Web Consortium. 2003. <http://www.w3.org/TR/2003/REC-PNG-20031110>
- 20 RFC 2045, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message*  
21 *Bodies.* Borenstein, N., and N. Freed. The Internet Society. 1996.  
22 <http://www.ietf.org/rfc/rfc2045.txt>.
- 23 RFC 2119 — *Key words for use in RFCs to Indicate Requirement Levels.* Bradner, S. The  
24 Internet Society. 1997. <http://www.rfc-editor.org>
- 25 RFC 3066 — *Tags for the Identification of Languages.* Alvestrand, H. The Internet Society. 2001.  
26 <http://www.rfc-editor.org>
- 27 RFC 4234 — *Augmented BNF for Syntax Specifications: ABNF.* Crocker, D. (editor). The Internet  
28 Society. 2005. <http://www.rfc-editor.org>
- 29 *A Standard Default Color Space for the Internet—sRGB, Version 1.10.* Anderson, Matthew,  
30 Srinivasan Chandrasekar, Ricardo Motta, and Michael Stokes. World Wide Web Consortium.  
31 1996. <http://www.w3.org/Graphics/Color/sRGB>
- 32 *TIFF, Revision 6.0.* Adobe Systems Incorporated. 1992.  
33 <http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf>
- 34 *Unicode Character Database, Revision 4.0.0.* Davis, Mark and Ken Whistler. The Unicode  
35 Consortium. 2003. <http://www.unicode.org/Public/4.0-Update/UCD-4.0.0.html>
- 36 *The Unicode Standard, Version 4.0.* The Unicode Consortium. Boston, MA: Addison-Wesley,  
37 2003, ISBN 0-321-18578-1.
- 38 *XML Base.* Marsh, Jonathan. World Wide Web Consortium. 2001.  
39 <http://www.w3.org/TR/2001/REC-xmlbase-20010627/>



- 1 *XML Schema Part 1: Structures, Second Edition*. Beech, David, Murray Maloney, Noah Mendelsohn, and Henry S. Thompson (editors). World Wide Web Consortium. 2004.
- 2 <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>
- 3
- 4 *XML Schema Part 2: Datatypes, Second Edition*. Biron, Paul V. and Ashok Malhotra (editors). World Wide Web Consortium. 2004. <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>
- 5



## 4. Definitions

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

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

**consumer** — A piece of software or a device that interprets and processes OpenXPS packages.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

- 1 **piece** — A portion of a part. Pieces of different parts can be interleaved together. The individual  
2 pieces are named using a unique mapping from the part name. Pieces are not addressable in  
3 the package model.
- 4 **primary fixed payload root** — The fixed payload root that is referenced by the OpenXPS  
5 package StartPart relationship.
- 6 **PrintTicket part** — A PrintTicket part provides the settings used when a package is printed.  
7 PrintTicket parts can be attached to the entire package, or at lower levels in the structure, such  
8 as individual pages.
- 9 **producer** — A piece of software or a device that creates or modifies OpenXPS packages.
- 10 **producer bleed size** — Represents the overflow (or “bleed”) box used by the producer for  
11 registration and layout.
- 12 **producer content size** — Represents the content bounding box specified by the producer.
- 13 **producer media size** — Represents the physical media on which the content will be printed.
- 14 **property** — A characteristic of a markup element, referred to as an attribute of the element.
- 15 **property attribute** — An OpenXPS Document property value can be expressed as either a  
16 property attribute or a property element.
- 17 **property element** — An OpenXPS Document property value can be expressed as either a  
18 property attribute or a property element.
- 19 **property value** — The value of a property, expressed as an XML attribute, an XML child  
20 element, or an entry in the resource dictionary.
- 21 **relationships** — A relationship represents the kind of connection between a source part and a  
22 target part in a package. Relationships make the connections between parts directly  
23 discoverable without looking at the content in the parts, and without altering the parts  
24 themselves. See also, package relationship.
- 25 **relationships part** — A part containing an XML representation of relationships.
- 26 **required part** — A part, such as an image or font, that is referenced from other parts, and is  
27 required for valid processing of the referencing part.
- 28 **resource definition** — A shareable property value, with a name, defined within a resource  
29 dictionary. Any property value defined by fixed page markup can be held in a resource  
30 dictionary. Each resource definition has a key that is unique within the scope of the resource  
31 dictionary.
- 32 **resource dictionary** — A resource dictionary holds resources. Each resource in a resource  
33 dictionary carries a name. The resource’s name can be used to reference the resource from a  
34 property’s XML attribute.
- 35 **resource dictionary, remote** — A part containing a resource dictionary.
- 36 **resource reference** — An attribute whose value refers to an entry in a resource dictionary.  
37 Resource references appear in the format “{StaticResource RscName}” where RscName  
38 corresponds to a matching entry in the resource dictionary with an x:Key attribute value.

- 1 **signature definition** — The means by which OpenXPS Document authors provide co-signature  
2 requirements and workflow-specific signature information.
- 3 **signature spot** — A visual element that indicates that a digital signature has been applied or  
4 requested.
- 5 **signing rules** — The set of rules that define whether a particular digital signature is compliant  
6 with the OpenXPS Document **signature** policy.
- 7 **story** — A block of individually readable content in an OpenXPS Document.
- 8 **story fragment** — A portion of a story that appears within the scope of a single fixed page.
- 9 **stream** — A linearly ordered sequence of bytes.
- 10 **thumbnail** — An images that helps end-users identify parts of a package or a package as a  
11 whole.
- 12 **X3D** — A 3D graphic content stream conforming to ISO standards 19775-1r1:200x, 19776-  
13 1:2005, 19776-2:2005, and 19776-3:2007.
- 14 **ZIP Archive** — A physical ZIP file that is displayed by the file system. A ZIP archive contains  
15 **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*.



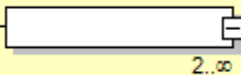
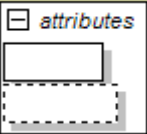


Syntax descriptions and code are formatted in `monospace` type.

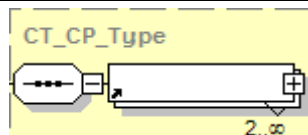
Replaceable items are formatted in *monospace cursive* type.

### 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.

12

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 Standard:

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

5 This Standard is divided into the following subdivisions:

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

20 Clauses 1–5 and 7–19, and annexes A, C–E and G, form a normative part of this Standard; and  
21 the clause 6, annexes B and F, 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. OpenXPS Document Format

This Standard describes how the OpenXPS Document format is organized internally and rendered externally. It is built upon the principles described in the OPC Standard. OpenXPS Documents MUST observe all conformance requirements [M1.1] and SHOULD observe all recommendations [S1.1] of that Standard, except where indicated otherwise. The information presented here is intended both for producers and consumers.

The OpenXPS 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 OpenXPS Document format implements the common package features specified by the OPC Standard that support digital signatures and core properties. Implementers should note that the OpenXPS Document format does not define support for encryption, or other forms of content protection, other than that required for Embedded Font Obfuscation.

---

### 8.1 How This Standard Is Organized

**This subclause is informative**

Clause	Description
Parts and Relationships (§9)	<p>This clause describes how OpenXPS Documents use the packaging model (as described in the OPC Standard) 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 OpenXPS 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 OpenXPS 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>

---

Clause	Description
Graphics (§11)	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.
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)	OpenXPS 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)	<p>This clause describes the components of the OpenXPS 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.</p> <p>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.</p>
OpenXPS Document Package Features (§17)	This clause describes how package features (as described in the OPC Standard) are used and extended in the OpenXPS Document format. This includes interleaving, digital signatures, and core properties.
Rendering Rules (§18)	This clause provides precise instructions for rendering OpenXPS 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.1)	This annex includes the W3C XSD schema for the Signature Definitions part.
OpenXPS Document Schema (§A.2)	This annex includes the W3C XSD schema for the FixedDocument, FixedDocumentSequence, and FixedPage parts.







## 9. Parts and Relationships

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

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

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

### 9.1 Fixed Payload

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

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

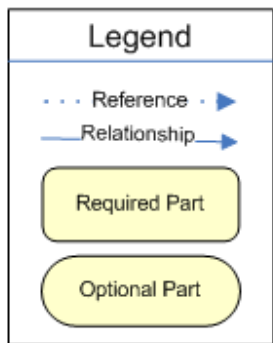
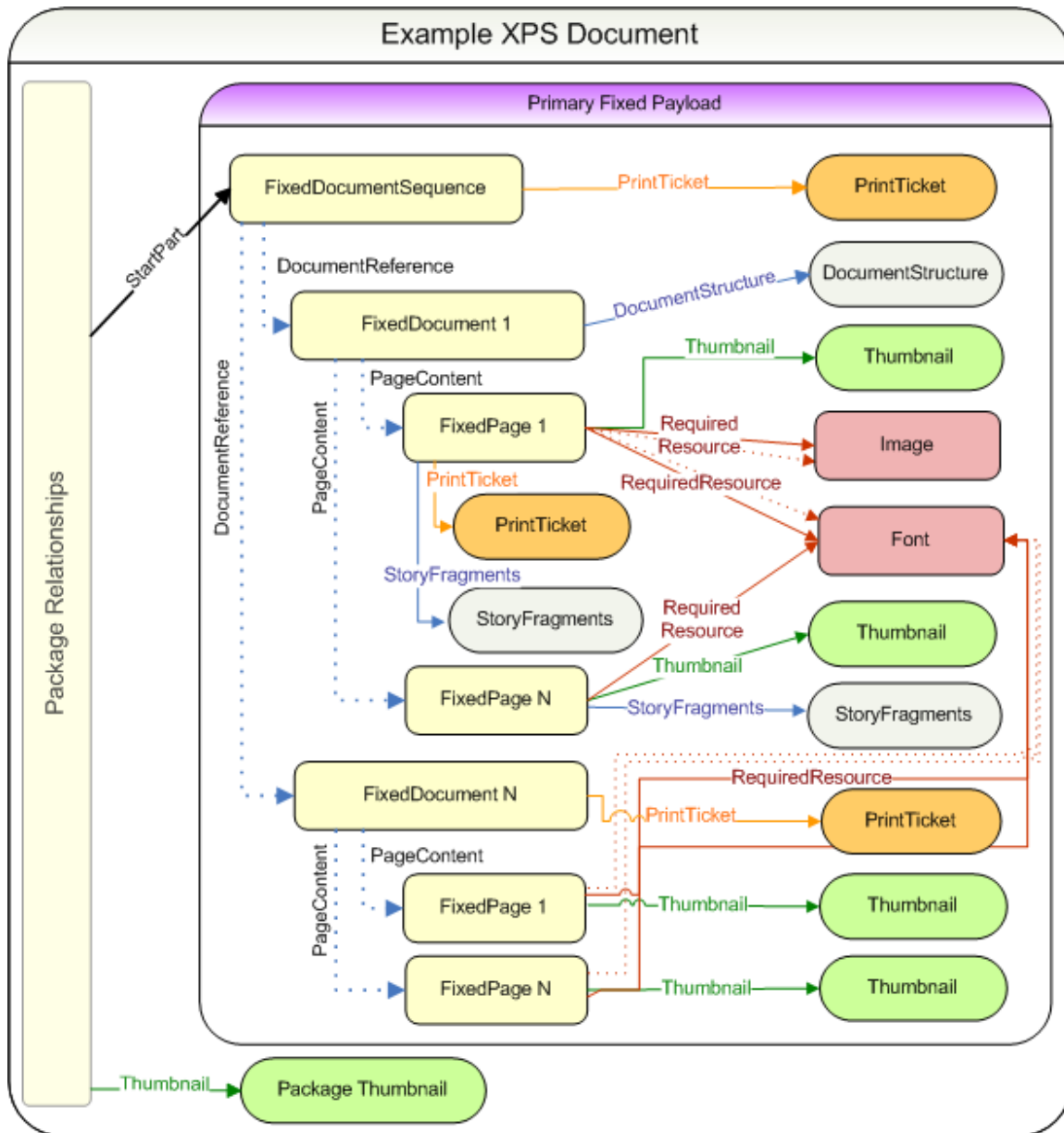
The payload includes the full set of parts required for processing the FixedDocumentSequence part. All content to be rendered MUST be contained in the OpenXPS Document [M2.1]. The payload ~~containing~~ for an OpenXPS Document may include additional parts not defined by this Standard. Consumers MUST ignore parts in valid OpenXPS Documents that they do not understand [M2.84]. The parts that can be found in an OpenXPS Document are listed in Table 9–1. Relationships and content types for these parts are defined in §D.2A. Each OpenXPS Document part MUST use *only* the appropriate content type specified in §D.2A [M2.2].

Table 9–1. OpenXPS 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)	<del>Contains an OpenType or TrueType</del>	REQUIRED if a

	<del>font</del> : <a href="#">Contains a font in the Open Font Format</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) <del>Windows Media Photo</del> <a href="#">JPEG XR</a> 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 color profile.	OPTIONAL [O2.3]
DocumentStructure (§9.1.11)	Contains the document outline and document contents (story definitions) for the OpenXPS 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 OpenXPS Document*



2  
3 *end example]*

### 9.1.1 Fixed Payload Relationships

Internal resources are associated with parts by relationships and inline references. OpenXPS Documents MUST NOT reference external OpenXPS 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 Standard.

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 §A.

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	REQUIRED for each

	to a required resource, including Font, Image, ColorProfile, and Remote Resource Dictionary parts. Required resources can be shared between pages.	resource referenced 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 OpenXPS Document  
2 for 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 instantiate an error condition if the part is not included in the OpenXPS 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 OpenXPS  
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 [M2.15]. Each  
17 <DocumentReference> element MUST reference a FixedDocument part by relative URI [M2.80].  
18 For more information, see §10.1.

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

### 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.

1 The markup in the FixedDocument part specifies the pages of a document in sequence using  
2 <PageContent> elements. The order of <PageContent> elements determines page order and  
3 MUST be preserved [M2.16]. Each <PageContent> element MUST reference a FixedPage part by  
4 relative URI [M2.81]. For more information, see §10.2.

5 The content type of the FixedDocument part is defined in §A,

#### 6 **9.1.4 FixedPage Part**

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

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

17 The content type of the FixedPage part is defined in §A

#### 18 **9.1.5 Image Parts**

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

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

26 OpenXPS Documents support the following image formats:

- 27 • JPEG
- 28 • PNG
- 29 • TIFF
- 30 • ~~Windows Media Photo~~ [JPEG XR](#)

31 Color profiles MAY be embedded in image files [O2.18]. See §15.

32 For images that have a constant opacity, producers SHOULD NOT use the image format alpha  
33 channel; the Opacity attribute in the <ImageBrush> element SHOULD be used instead [S2.37].

##### 34 **9.1.5.1 JPEG Images**

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

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

<b>APPn marker</b>	<b>Originating source</b>
APP0	JFIF specification
APP1	EXIF extension defined by JEITA
APP2	ICC profile marker defined by the ICC specification

2 Consumers MUST ensure that they can distinguish between the uses of those markers listed in  
3 Table 9–3 and other data that is recorded using the same markers [M2.85].

4 [*Note:* The APP0 marker is also used for JFXX (extended JFIF). The APP1 marker is also used  
5 for XMP metadata. The APP2 marker is also used for EXIF FlashPix extensions. These are not  
6 intended to be exhaustive lists of alternative uses of those markers. *end note*]

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

10 In cases where a consumer encounters a JPEG image with conflicting resolution information in  
11 different markers, the order of precedence is as follows:

- 12 1. The EXIF tag
- 13 2. The JFIF tag
- 14 3. Any other APPn tags supported by the consumer
- 15 4. A default value of 96 dots per inch (dpi) (as described in §13.4.1)

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

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

21 Therefore, the use of JPEG CMYK images is NOT RECOMMENDED in OpenXPS Documents  
22 because rendering results can differ significantly between implementations. TIFF or ~~Windows~~  
23 ~~Media-Photo~~[JPEG XR](#) images SHOULD be used instead to represent CMYK images [S2.7].

24 If both ICC-specified APP2, and APP13 markers are specified, the ICC-specified APP2 marker  
25 takes precedence. If the JPEG image is embedded in a TIFF image, the TIFF ICC profile settings  
26 are used.

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

### 30 **9.1.5.2 PNG Images**

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

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

<b>Chunk</b>	<b>Support Level</b>
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]

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

### 5 **9.1.5.3 TIFF Images**

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

11 *Table 9–5. Supported TIFF tags*

<b>Image type</b>	<b>Tags</b>
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, or 32773) ImageLength ImageWidth ResolutionUnit (1, 2, or 3) RowsPerStrip StripByteCounts StripOffsets XResolution YResolution
Palette color	BitsPerSample (1, 4, or 8)



---

images	ColorMap Compression (1, 5, or 32773) ImageLength ImageWidth PhotometricInterpretation (3) ResolutionUnit (1, 2, or 3) RowsPerStrip StripByteCounts StripOffsets XResolution YResolution
RGB images	BitsPerSample (8,8,8 or 16,16,16; <i>or if SamplesPerPixel = 4: 8,8,8,8 or 16,16,16,16</i> ) Compression (1, 5, 7, or 32773) ExtraSamples (0, 1, or 2. Required if SamplesPerPixel = 4; must not be present otherwise) ICC Color Profile [tag 34675] ImageLength ImageWidth PhotometricInterpretation (2) PlanarConfiguration (1) ResolutionUnit (1, 2, or 3) RowsPerStrip SamplesPerPixel (3 or 4) StripByteCounts StripOffsets XResolution YResolution

---

---

CMYK images (TIFF extension)	BitsPerSample (8,8,8,8 or 16,16,16,16; <i>or</i> if SamplesPerPixel = 5: 8,8,8,8,8 or 16,16,16,16,16) Compression (1, 5, 7, or 32773) ExtraSamples (0, 1, or 2. Required if SamplesPerPixel = 5; must not be present otherwise) ICC Color Profile [tag 34675] ImageLength ImageWidth InkSet (1) NumberOfInks (4) PhotometricInterpretation (5) PlanarConfiguration (1) ResolutionUnit (1, 2, or 3) RowsPerStrip SamplesPerPixel (4 or 5) StripByteCounts StripOffsets XResolution YResolution
------------------------------------	---

---

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

3 If the ResolutionUnit tag is set to 1 (no units), XResolution and YResolution are interpreted in  
4 the same manner as if the ResolutionUnit was set to 2 (inches).

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

8 The following features of the TIFF specification MUST be supported in addition to the tags  
9 described in Table 9–5:

- 10 • Baseline TIFF (Sections 1–10) with the exception of the following tags [M2.26]:
  - 11 ○ CellLength
  - 12 ○ CellWidth
  - 13 ○ GrayResponseCurve
  - 14 ○ GrayResponseUnit
  - 15 ○ MaxSampleValue
  - 16 ○ MinSampleValue
  - 17 ○ Orientation
  - 18 ○ Thresholding
- 19 • CCITT bilevel encodings (Section 11) [M2.28]
- 20 • CMYK images (Section 16) [M2.29]
- 21 • Associated alpha data (Section 18) [M2.30]

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

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

16 OpenXPS Document consumers SHOULD mitigate the effect of badly formed TIFF files in the  
17 following ways [S2.11]:

- 18      • Test with as many different TIFF images as possible.
- 19      • Correct common mistakes in TIFF images, such as:
  - 20       ○ Not all BitsPerSample hold the same value
  - 21       ○ Number of BitsPerSample does not match SamplesPerPixel
  - 22       ○ PhotometricInterpretation 1 or 2 (instead of 3) used when BitsPerSample is set to
  - 23       "8,8,8"
  - 24       ○ When the ExtraSamples tag is missing and SamplesPerPixel is not consistent with the
  - 25       PhotometricInterpretation tag then ExtraSamples values should be given the value 0.
- 26      • Implement a recovery strategy when a problematic TIFF image is encountered.

27 [*Note*: Over time, TIFF-consuming implementations have developed a certain tolerance for such  
28 deviations by attempting to deduce the intent of the TIFF image author and correct for  
29 apparent errors or deviations.

30 Many TIFF images in circulation today deviate from the TIFF Specification. *end note*]

#### 31 **9.1.5.4 ~~Windows Media Photo~~JPEG XR Images**

32 It is RECOMMENDED that ~~Windows Media Photo~~JPEG XR image part names end with the  
33 extension ".wdpjxr" [S2.12]. ~~Windows Media Photo~~JPEG XR image ~~parts files~~-MUST conform to  
34 the ~~Windows Media Photo~~JPEG XR specification [M2.35] and MUST use the Tag-based file  
35 format defined in Annex A of the JPEG XR specification [M2.91]. OpenXPS Documents support  
36 ~~Windows Media Photo~~JPEG XR images with the characteristics identified in Table 9–6 and §15.3.

37 *Table 9–6. Supported ~~Windows Media Photo~~JPEG XR features*

Color space	Pixel formats	Compression	Alpha
Grayscale	<a href="#">BlackWhite</a>	Lossy	None
		<u>- or -</u>	

	8-bit integer	Lossless	
	16-bit integer		
	16-bit half-float*		
	16-bit fixed point*		
	32-bit fixed point*		
sRGB	8-bit integer	Lossy	1-channel
	16-bit integer	<u>- or -</u>	<u>- or -</u>
		Lossless	1-channel pre-multiplied
scRGB	16-bit half-float		1-channel
	16-bit fixed point	Lossy	<u>- or -</u>
	32-bit IEEE float	<u>- or -</u>	1-channel pre-multiplied
	32-bit fixed point	Lossless	RGBE-Radiance (no alpha channel)
	RGBE-Radiance		
CMYK	8-bit integer	Lossy	
	16-bit integer	<u>- or -</u>	1-channel independent
		Lossless	
N-channel (including named color N-tone)	8-bit integer	Lossy	
	16-bit integer	<u>- or -</u>	1-channel independent
		Lossless	
Profiled RGB (3-channel)	8-bit integer	Lossy	1-channel
	16-bit integer	<u>- or -</u>	<u>- or -</u>
		Lossless	1-channel pre-multiplied

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

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

### 6 **9.1.6 Thumbnail Parts**

7 Thumbnails are small images that represent the contents of a fixed page or an entire OpenXPS  
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 §A. 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 Standard allows thumbnails to be attached to any part, OpenXPS 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 OpenXPS Document consumers [S2.14]. The content types of thumbnail parts  
18 are specified in §D.2.

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

### 9.1.7 Font Parts

Fonts are stored in font parts. OpenXPS Documents MUST support the OpenType font format (ISO/IEC 14496-22:2007), ~~which includes~~ TrueType and CFF fonts [M2.39M2.39]. To support portability, Unicode-encoded fonts SHOULD be used (see §9.1.7.5 for additional information) [S2.15].

[\[Note: The Open Font Format is considered to be equivalent to the OpenType font format. end note\]](#)

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 instantiate an error condition [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 §A.

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

#### 9.1.7.1 Subsetting Fonts

OpenXPS 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 Format~~ font file [M2.39]. Requirements for valid Open ~~Font Format~~ font files are described in the Open ~~Font Format~~ Font File specification.

#### 9.1.7.2 Open ~~Font Format~~ Font Embedding

Protecting the intellectual property of font vendors is a goal of the OpenXPS Document format. Therefore, producers MUST observe the guidelines and mechanisms described below in order to honor the licensing rights specified in Open ~~Font Format~~ fonts [M2.40]. It is not the responsibility of consumers to enforce font licensing intent, although consumers MUST be able to process OpenXPS 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 Open ~~Font Format~~ 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

1 intents and requirements or recommendations. These requirements represent the “rules” that  
2 producers and consumers must follow in order to respect licensing rights specified in the font.

3 For further details on licensing rights of Open [Font FormatType fonts](#), see the description of the  
4 OS/2 table in “OS/2 and Windows Metrics.”

5 *Table 9–7. Guidelines for Open [Font FormatType 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 §D.3 for details. MUST NOT modify or edit the OpenXPS Document markup or hierarchical structure starting from the <FixedDocument> element [M2.43].	MUST NOT extract or install permanently [M2.45].
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	MUST NOT extract or install permanently

		(see §9.1.7.3) [M2.48]. [M2.50]. MUST NOT subset font before embedding. [M2.49]	
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 are present in the font, MUST NOT embed the font [M2.51].	MUST NOT extract or install permanently [M2.52].
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 OpenXPS 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 OpenXPS 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  
17 OpenXPS Documents [S2.19]. However, there are vertical solutions in which implementations  
18 might 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<sub>00</sub>, B<sub>01</sub>,  
25 B<sub>02</sub>, B<sub>03</sub>; B<sub>10</sub>, B<sub>11</sub>; B<sub>20</sub>, B<sub>21</sub>; B<sub>30</sub>, B<sub>31</sub>, B<sub>32</sub>, B<sub>33</sub>, B<sub>34</sub>, B<sub>35</sub>, B<sub>36</sub>, and B<sub>37</sub>. 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.

2. Generate a part name for the obfuscated font using the GUID. The last segment of the 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 " $B_{03}B_{02}B_{01}B_{00}-B_{11}B_{10}-B_{21}B_{20}-B_{30}B_{31}-B_{32}B_{33}B_{34}B_{35}B_{36}B_{37}.ext$ " where each  $B_x$  represents a placeholder for one byte of the GUID, represented as two hex digits [M2.54]. The part name MAY have an arbitrary extension (identified by the placeholder ".ext") [O2.22]. It is RECOMMENDED that the extension for TrueType fonts be ".odttf" and for TrueType collections be ".odttc" [S2.20].
3. The content type for the part containing the obfuscated font MUST match the definition in §A [M2.2].
4. Perform an XOR operation on the first 32 bytes of the binary data of the font part with the array consisting of the bytes referred to by the placeholders  $B_{37}$ ,  $B_{36}$ ,  $B_{35}$ ,  $B_{34}$ ,  $B_{33}$ ,  $B_{32}$ ,  $B_{31}$ ,  $B_{30}$ ,  $B_{20}$ ,  $B_{21}$ ,  $B_{10}$ ,  $B_{11}$ ,  $B_{00}$ ,  $B_{01}$ ,  $B_{02}$ , and  $B_{03}$ , in that order and repeating the array once. The result is an obfuscated font.
5. Store the obfuscated font in a part with the generated name.

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

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

#### 9.1.7.4 Print and Preview Restricted Fonts

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

When editing content, producers MUST NOT edit a document where the FixedDocument part has a Restricted Font relationship [M2.43]. When editing content, producers MUST instantiate an error condition when encountering any font with the print and preview restriction bit set for which no Restricted Font relationship has been added to the FixedDocument part [M2.12]. Consumers that are not also producers MUST consider an OpenXPS Document valid even if the producer failed to properly set the Restricted Font relationship [M2.12].

#### 9.1.7.5 Non-Standard Font Compatibility Encoding

When processing <Glyphs> elements, the consumer MUST first select a cmap table from the Open [Font Format](#) ~~Type font~~ following the order of preference shown below (highest listed first) [M2.55]:



1 *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

2 All further processing for that font MUST use the selected cmap table [M2.55].

3 If a Wansung, Big5, Prc, ShiftJis or MacRoman cmap has been selected, the consumer MUST  
 4 correctly map from Unicode codepoints in the UnicodeString to the corresponding codepoints  
 5 used by the cmap before looking up the glyphs [M2.56]. The Unicode standard provides details  
 6 of the required mappings.

7 Producers SHOULD avoid using fonts lacking a Unicode-encoded cmap table [S2.15].

8 When processing <Glyphs> elements that reference a cmap (3,0) encoding font, consumers  
 9 MUST be prepared for the case in which the UnicodeString attribute contains character codes  
 10 instead of PUA codepoints [M2.57]. This condition is indicated by an unsuccessful Unicode  
 11 lookup of the codepoint specified in the Unicode string in the cmap (3,0) table. In this case, the  
 12 correct glyph index is computed by following the general recommendations of the Open [Font](#)  
 13 [FormatType](#) specification.

14 When processing <Glyphs> elements that use this compatibility encoding, character codes in  
 15 the range 0x20-0xff are mapped to PUA codepoints. [See §12.1.4 for requirements for handling](#)  
 16 [Unicode control marks](#). ~~Therefore, character codes in the range 0x80-0x9f are not considered~~  
 17 ~~non-printable Unicode control codes.~~

18 This non-standard encoding has been included to facilitate document production for certain  
 19 producers. However, there are significant drawbacks resulting from this encoding:

- 20 • Search is unpredictable
- 21 • Copy and paste functionality is unpredictable
- 22 • Glyph rendering is unpredictable, especially between different consumers

23 Producers SHOULD NOT use this non-standard encoding and they SHOULD write PUA  
 24 codepoints to the UnicodeString attribute [S2.15].

### 25 **9.1.8 Remote Resource Dictionary Parts**

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

### 9.1.9 PrintTicket Parts

This Standard provides a mechanism for including user intent and device configuration settings within an OpenXPS Document as PrintTicket parts. *PrintTicket parts* enable the association of settings with parts within an OpenXPS Document. The format to be used for PrintTickets is implementation-defined. This Standard defines how to associate those PrintTicket parts with OpenXPS Documents. If the consumer understands the content of the PrintTicket, then the PrintTicket part SHOULD be processed when the OpenXPS Document is printed [S2.36]. PrintTicket parts can be attached only to FixedDocumentSequence, FixedDocument, and FixedPage parts, and each of these parts MUST attach no more than one PrintTicket [M2.59].

#### 9.1.9.1 Mapping PrintTicket Parts to Fixed Payload Parts

OpenXPS Documents contain a hierarchy of FixedDocumentSequence, FixedDocument, and FixedPage parts, as defined in §10. The association of PrintTickets with FixedDocumentSequence, FixedDocument, and FixedPage parts reflects this hierarchy and enables the scope of settings specified in PrintTicket parts to be limited to the FixedDocumentSequence, FixedDocument, and FixedPage parts within the OpenXPS Document. Domain-specific implementations are responsible for specifying how the settings provided in the PrintTicket parts are scoped.

### 9.1.10 SignatureDefinitions Part

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

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

The SignatureDefinitions part is OPTIONAL [O2.6]. Signature definitions MUST conform to the Signature Definitions schema as defined in §A.1 [M2.86].

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

### 9.1.11 DocumentStructure Part

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

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

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

Consumers MAY provide an algorithmic construction of the structure of an OpenXPS Document based on a page-layout analysis [O2.27], but they MUST NOT use such a method to derive

1 structure for any part of the OpenXPS Document included in the DocumentStructure part  
2 [M2.68]. A consumer capable of calculating reading order from the layout of the document  
3 MUST use the reading order specified in the DocumentStructure part, even though the derived  
4 order might be perceived as preferable to the specified order [M2.68].

### 5 **9.1.12 StoryFragments Part**

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

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

11 The StoryFragments part is referenced by relationship from its associated FixedPage part. The  
12 content type of the StoryFragments part is specified in §D.2.

---

## 13 **9.2 Part Naming Recommendations**

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

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

22 It is RECOMMENDED that producers of OpenXPS Documents use the following part naming  
23 convention:

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

- 1 A Font part name SHOULD append the segment "Fonts/" to the resource part name prefix  
 2 specified above [S2.27]. [*Example*: A font might be named  
 3 "/Documents/1/Resources/Fonts/Arial.ttf" or "/Resources/Fonts/F2ABC7B7-C60D-4FB9-  
 4 AAE4-3CA0F6C7038A.odttf". *end example*]
- 5 An Image part name SHOULD append the segment "Images/" to the resource part name  
 6 specified above [S2.27]. [*Example*: An image might be named  
 7 "/Documents/3/Resources/Images/dog.jpg" or "/Resources/Images/E0D79307-846E-  
 8 11CE-9641-444553540000.jpg". *end example*]
- 9 A Remote Resource Dictionary part name SHOULD append the segment "Dictionaries/" to  
 10 the resource part name specified above [S2.27]. Remote resource dictionaries SHOULD  
 11 also use the ".dict" extension [S2.27]. [*Example*: A resource dictionary might be named  
 12 "/Documents/2/Resources/Dictionaries/Shapes.dict" or  
 13 "/Resources/Dictionaries/0DDF3BE2-E692-15D1-AB06-B0AA00BDD685.dict". *end*  
 14 *example*]
- 15 • Any DocumentStructure part name SHOULD contain four segments using  
 16 "/Documents/*n*/Structure/" as the first three segments and the extension ".struct"  
 17 [S2.28]. Here *n* represents the fixed document that this structure is associated with.  
 18 [*Example*: The DocumentStructure part for the first document in a fixed document  
 19 sequence could be "/Documents/1/Structure/DocStructure.struct". *end example*]
  - 20 • Any StoryFragments part name SHOULD contain five segments using  
 21 "/Documents/*n*/Structure/Fragments" as the first four segments and the extension  
 22 ".frag" [S2.29]. Here *n* represents the fixed document that these parts are associated  
 23 with. [*Example*: A StoryFragment part associated with the third page of the second  
 24 document in a fixed document sequence could be  
 25 "/Documents/2/Structure/Fragments/3.frag". *end example*]
  - 26 • ICC profile part names SHOULD contain four segments, using "/Documents/*n*/Metadata/"  
 27 as the first three segments, where *n* is the fixed document that uses these parts  
 28 [S2.30]. If an ICC profile part is shared across documents, the part name SHOULD  
 29 contain two segments, using "/Metadata/" as the first segment and a second segment  
 30 that is a string representation of a globally unique identifier, followed by an extension  
 31 [S2.30]. ICC profiles SHOULD use an appropriate extension for the color profile type.  
 32 [S2.30] [*Example*: ".icm" *end example*]
  - 33 • Thumbnail part names SHOULD contain four segments, using "/Documents/*n*/Metadata/"  
 34 as the first three segments, where *n* is the fixed document that uses the thumbnail  
 35 [S2.31]. If the Thumbnail part relates to the package as a whole, the part name  
 36 SHOULD contain two segments, using "/Metadata/" as the first segment and a second  
 37 segment that is a string representation of a globally unique identifier, followed by an  
 38 extension [S2.31]. Thumbnails SHOULD use an extension appropriate to the image type,  
 39 either ".png" or ".jpg" [S2.31]. [*Example*: A Thumbnail part for a particular fixed page  
 40 might be "/Documents/1/Metadata/5.png". *end example*]
  - 41 • PrintTicket part names associated with the entire job SHOULD be associated via  
 42 relationship with the FixedDocumentSequence part and contain two segments, using  
 43 "/Metadata/" as the first segment [S2.32]. PrintTicket parts associated with a particular  
 44 fixed document or fixed page SHOULD contain four segments, using  
 45 "/Documents/*n*/Metadata/" as the first three segments, where *n* is the fixed document  
 46 that uses these parts [S2.32]. PrintTicket parts based on XML SHOULD use the  
 47 extension ".xml" [S2.32]. [*Example*: A PrintTicket associated with the entire job could be  
 48 "/Metadata/Job\_PT.xml" and a PrintTicket associated with a single page might be  
 49 "/Documents/1/Metadata/Page2\_PT.xml". *end example*]

- The names of any non-standard parts that are associated with a particular fixed document SHOULD contain four segments, using “/Documents/*n*/Other/” as the first three segments. Here, *n* is the fixed document to which the part belongs [S2.33].

4 *Example 9–2. OpenXPS Document part naming*

5 An OpenXPS Document that contains two FixedDocument parts is represented as follows:

```
6 /FixedDocSeq.fdseq
7 /Documents/1/FixedDocument.fdoc
8 /Documents/1/Pages/1.fpage
9 /Documents/1/Pages/2.fpage
10 /Documents/1/Resources/Fonts/FontA.ttf
11 /Documents/1/Resources/Images/ImageB.jpg
12 /Documents/1/Metadata/Document_PT.xml
13 /Documents/1/Metadata/Page5_PT.xml
14 /Documents/1/Structure/DocStructure.struct
15 /Documents/1/Structure/Fragments/1.frag
16 /Documents/1/Structure/Fragments/2.frag
17 /Documents/1/Other/FabrikamIncBussinessAccount.xml
18 /Documents/2/FixedDocument.fdoc
19 /Documents/2/Pages/1.fpage
20 /Documents/2/Resources/Fonts/FontB.ttf
21 /Documents/2/Resources/Images/ImageA.png
22 /Documents/2/Metadata/ColorProfile.icm
23 /Documents/2/Metadata/Document_PT.xml
24 /Documents/2/Other/FabrikamIncInsuranceInfo.xml
25 /Metadata/Job_PT.xml
26 /Resources/Fonts/63B51F81-C868-11D0-999C-00C04FD655E1.ttf
```

27 *end example]*

---

### 28 **9.3 OpenXPS Document Markup**

29 OpenXPS Document Markup is used to describe the content of fixed pages within an OpenXPS  
 30 Document. This XML-based markup has been designed to address the requirements for  
 31 describing graphical content within electronic paper documents. The graphical primitives  
 32 described by the elements, attributes and attribute values in the markup are completely  
 33 sufficient for representing document content as acquired from, or output to, physical paper by a  
 34 variety of document devices and applications. The OpenXPS Document Markup has also been  
 35 developed consistent with the independent development of compatible systems that produce or  
 36 consume OpenXPS Documents.

37 The design of OpenXPS Document Markup reflects the tradeoffs between the following two,  
 38 sometimes competing, goals:

1. OpenXPS Document markup should be parsimonious; that is, it should include only the  
 40 minimum set of primitive operations and markup constructs necessary to render text and  
 41 graphics with full fidelity. Redundancy in the Standard increases the opportunity for  
 42 independent implementations, such as printer-resident raster image processors (RIPs),  
 43 viewers, and interactive applications, to introduce accidental incompatibilities. Redundancy  
 44 also increases the cost of implementation and testing, and, typically, the required memory  
 45 footprint.

2. OpenXPS Document markup should be compact; that is, the most common graphical primitives for vector graphics and text-rendering should have compact representations. Inefficient representations compromise the performance of systems handling OpenXPS Documents. As byte-count increases, so does communication time. Although compression can be used to improve communication time, it cannot eliminate the performance loss caused by inefficient representations.

### 9.3.1 Support for Versioning and Extensibility

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

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

- [DiscardControl](#)
- DocumentStructure
- FixedDocument
- FixedDocumentSequence
- FixedPage
- Relationships
- Remote Resource Dictionary
- SignatureDefinitions
- StoryFragments

Consumers of these parts MUST support the Markup Compatibility and Extensibility Standard [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 OpenXPS Document markup [M2.69].

Markup compatibility elements and attributes that appear in one OpenXPS 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 Standard 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 Standard), that declaration MUST NOT name any encoding other than UTF-8 or UTF-16 [M2.70]. [\[Note: This Standard specifies unambiguously how implementations should operate with XML content and does so in terms of UTF-16 encoding. Note that this does not preclude the use of UTF-8 in OpenXPS Document content. end note\]](#)
2. The XML 1.0 Standard 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 Standard, and consumers MUST instantiate an error condition when encountering DTD content [M2.71].

- 1       3. If the XML content contains the Markup Compatibility and Extensibility namespace, as  
2       described in the Markup Compatibility and Extensibility Standard, it MUST be processed  
3       to remove Markup Compatibility and Extensibility elements and attributes, ignorable  
4       namespace declarations, and ignored elements and attributes before applying further  
5       validation rules below [M2.69].
- 6       4. XML content MUST be valid against the corresponding W3C XSD schema defined in this  
7       Standard. In particular, the XML content MUST NOT contain elements or attributes drawn  
8       from namespaces that are not explicitly defined in the corresponding XSD unless the XSD  
9       allows elements or attributes drawn from any namespace to be present in particular  
10      locations in the XML markup [M2.72].
- 11     5. XML content MUST NOT contain elements or attributes drawn from "xml" or "xsi"  
12     namespaces unless they are explicitly defined in the W3C XSD schema or by other means  
13     in the Standard [M2.73].

### 14   **9.3.3 Markup Model**

15   OpenXPS Document markup is an XML-based markup language that uses elements, attributes,  
16   and namespaces. The schema for OpenXPS Document markup includes only elements and their  
17   attributes, comments, and whitespace. Arbitrary character data intermingled in the markup is  
18   not allowed.

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

23   OpenXPS Document markup also uses resources and resource dictionaries, which allow  
24   elements to share property values.

#### 25   **9.3.3.1 Namespaces**

26   The following XML namespaces are defined for use in OpenXPS Document markup:

- 27     • The OpenXPS Document namespace, the principal namespace used for elements and  
28     attributes in fixed page markup. For more information, see §A.
- 29     • The Resource Dictionary Key namespace, which allows certain OpenXPS Document  
30     elements to be included in a resource dictionary, as described in §14.2.
- 31     • The Markup Compatibility namespace, which supports the Markup Compatibility and  
32     Extensibility Standard as defined in the OPC Standard.

#### 33   **9.3.3.2 Properties**

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

38   Properties MUST NOT be set more than once, regardless of the syntax used to specify the value  
39   [M2.74]. In certain cases, they can be specified using either property attributes or property  
40   elements. Consumers MUST instantiate an error condition when encountering properties that  
41   are specified in both ways [M2.74].

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

#### 1 **9.3.3.2.1 Composable Property Values**

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

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

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

#### 13 **9.3.3.2.2 Property Attribute Syntax**

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

17 *Example 9-3. Property attribute syntax*

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

```
19 <!-- Property Attribute Syntax -->  
20 <SolidColorBrush Color="#FF0000" />
```

21 *end example*]

#### 22 **9.3.3.2.3 Property Element Syntax**

23 Some property values can also be expressed using a child element to describe the property  
24 value. These property elements are included to enable usage of the markup compatibility  
25 mechanisms described in the Markup Compatibility and Extensibility Standard. The element  
26 name is derived from a combination of a parent element name and the property name,  
27 separated by a dot (.) character.

28 The order of child property elements is significant: they **MUST** occur before any contents of the  
29 parent element and they **MUST** appear in the sequence specified in the schema [M2.87].



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 OpenXPS 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]. OpenXPS Document markup  
26 MUST NOT use the xml:space attribute [M2.75]. Additionally, where the OpenXPS Document  
27 schema specifies attributes of types that allow whitespace collapsing, leading and trailing  
28 whitespace in the attribute value MAY be used along with other whitespace that relies on the  
29 whitespace collapsing behavior specified in the XML Schema Standard [O2.32].

30 [*Note*: Consult the OpenXPS 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 OpenXPS Document, the language of the  
43 copied 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 OpenXPS Document MUST be identified using the xml:lang  
5 attribute, the value of which is inherited by child and descendant elements [M2.76].

6 This attribute is defined in the W3C XML Standard.

7 xml:lang is REQUIRED for <FixedPage> elements [M2.88]. xml:lang MAY be used with <Canvas>,  
8 <Path>, and <Glyphs> elements [O2.33]. xml:lang MUST NOT be used on any other fixed page  
9 markup element [M2.89]. xml:lang is also REQUIRED for the <DocumentOutline> element for  
10 document structure [M2.90]. xml:lang is OPTIONAL for the <OutlineEntry> element [O2.34].  
11 When the language of the contents is unknown and is required, the value "und" (undetermined)  
12 MUST be used [M2.76].

# 10. Documents

OpenXPS 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/2005/06schemas.openxps.org/oxps/v1.0">
  <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 OpenXPS</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 OpenXPS
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 OpenXPS								

	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].

## 10.2 <FixedDocument> Element

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
10 |   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">
11 |     <PageContent Source="Pages/1.fpage" />
12 |     <PageContent Source="Pages/2.fpage" />
13 |   </FixedDocument>
    
```

14 *end example]*

### 10.2.1 <PageContent> Element

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>URI</td> <td>Required</td> <td></td> <td>Yes</td> <td></td> </tr> <tr> <td>Width</td> <td>float</td> <td>Optional</td> <td></td> <td>Yes</td> <td></td> </tr> <tr> <td>Height</td> <td>float</td> <td>Optional</td> <td></td> <td>Yes</td> <td></td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	Annotation	Source	URI	Required		Yes		Width	float	Optional		Yes		Height	float	Optional		Yes	
Name	Type	Use	Default	Fixed	Annotation																				
Source	URI	Required		Yes																					
Width	float	Optional		Yes																					
Height	float	Optional		Yes																					

	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 OpenXPS Document [M3.5].
	Width	<u>ST_GEOne</u>				The width of the page contained in the page content.
	Height	<u>ST_GEOne</u>				The height of the page 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 [M3.21].

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 OpenXPS Document,  
 14 even in other fixed documents within the fixed payload [M3.6].

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

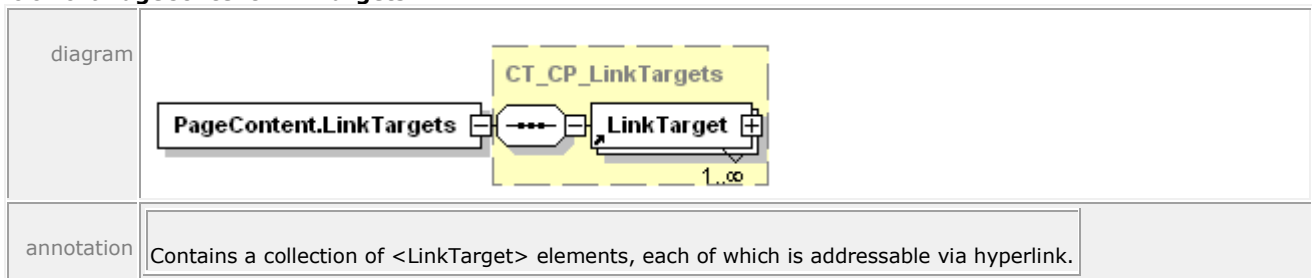
```

16 <FixedDocument
17   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">
18   <PageContent Source="Pages/1.fpage" Height="1056" Width="816" />
19   <PageContent Source="Pages/2.fpage" Height="1056" Width="816" />
20 </FixedDocument>
    
```

21 *end example]*

### 22 10.2.2 <PageContent.LinkTargets> Element

23 element **PageContent.LinkTargets**



24 The <PageContent.LinkTargets> element defines the list of link targets that specify each named  
 25 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
5 |   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">
6     <PageContent Source="Pages/1.fpage" Height="1056" Width="816" />
7     <PageContent Source="Pages/2.fpage" Height="1056" Width="816">
8         <PageContent.LinkTargets>
9             <LinkTarget Name="Anchor1" />
10            <LinkTarget Name="Anchor2" />
11        </PageContent.LinkTargets>
12    </PageContent>
13 </FixedDocument>
```

14 *end example]*

### 15 10.2.3 <LinkTarget> Element

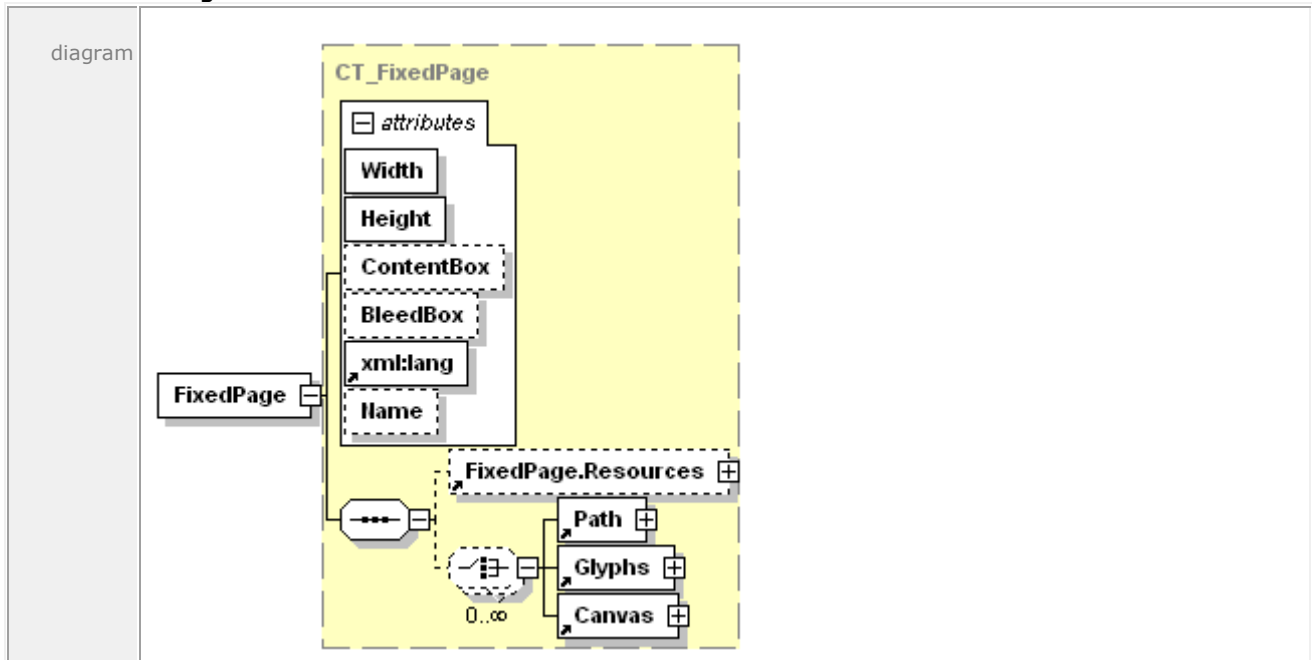
16 element **LinkTarget**

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.					

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

1 **10.3 <FixedPage> Element**

2 element **FixedPage**



attributes	Name	Type	Use	Default	Fixed	Annotation
		Width	<u>ST_GEOne</u>	required		
	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>				<del>Specifies the area including crop marks that extends outside of the physical page.</del> <a href="#">Specifies the union of the ContentBox and the bounding box of all graphical content intended to appear on the final printed and trimmed page.</a> 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 [M3.22].

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/2005/06schemas.openxps.org/oxps/v1.0">
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 BleedBox attribute defines the union of the ContentBox and the bounding  
 22 box of all graphical content intended to appear on the final printed and trimmed page. Workflow  
 23 artifacts such as crop marks are not normally intended to appear in the final page and do not  
 24 play a part in defining the size or position of the BleedBox.

25 The bleed box is expressed as four comma-separated, real-number coordinate values that  
 26 correspond to BleedOriginX, BleedOriginY, BleedWidth, BleedHeight. These values are specified  
 27 in units of 1/96".

28

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

- 31 • The BleedBox BleedOriginX value MUST be less than or equal to 0 [M3.7].
- 32 • The BleedBox BleedOriginY value MUST be less than or equal to 0 [M3.8].



- 1 • The BleedBox BleedWidth value MUST be greater than or equal to the fixed page Width  
2 attribute value plus the absolute value of the Bleedbox BleedOriginX value [M3.9].
- 3 • The BleedBox BleedHeight value MUST be greater than or equal to the fixed page Height  
4 attribute value plus the absolute value of the BleedBox BleedOriginY value [M3.10].

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

### 6 **10.3.2 ContentBox Attribute**

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

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

- 14 • The ContentBox ContentOriginX value MUST be greater than or equal to 0 and less than  
15 the fixed page Width attribute value [M3.11].
- 16 • The ContentBox ContentOriginY value MUST be greater than or equal to 0 and less than  
17 the fixed page Height attribute value [M3.12].
- 18 • The ContentBox ContentWidth value MUST be less than or equal to the difference between  
19 the fixed page Width attribute value and the ContentBox ContentOriginX value [M3.13].
- 20 • The ContentBox ContentHeight value MUST be less than or equal to the difference  
21 between the fixed page Height attribute value and the ContentBox ContentOriginY value  
22 [M3.14].

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

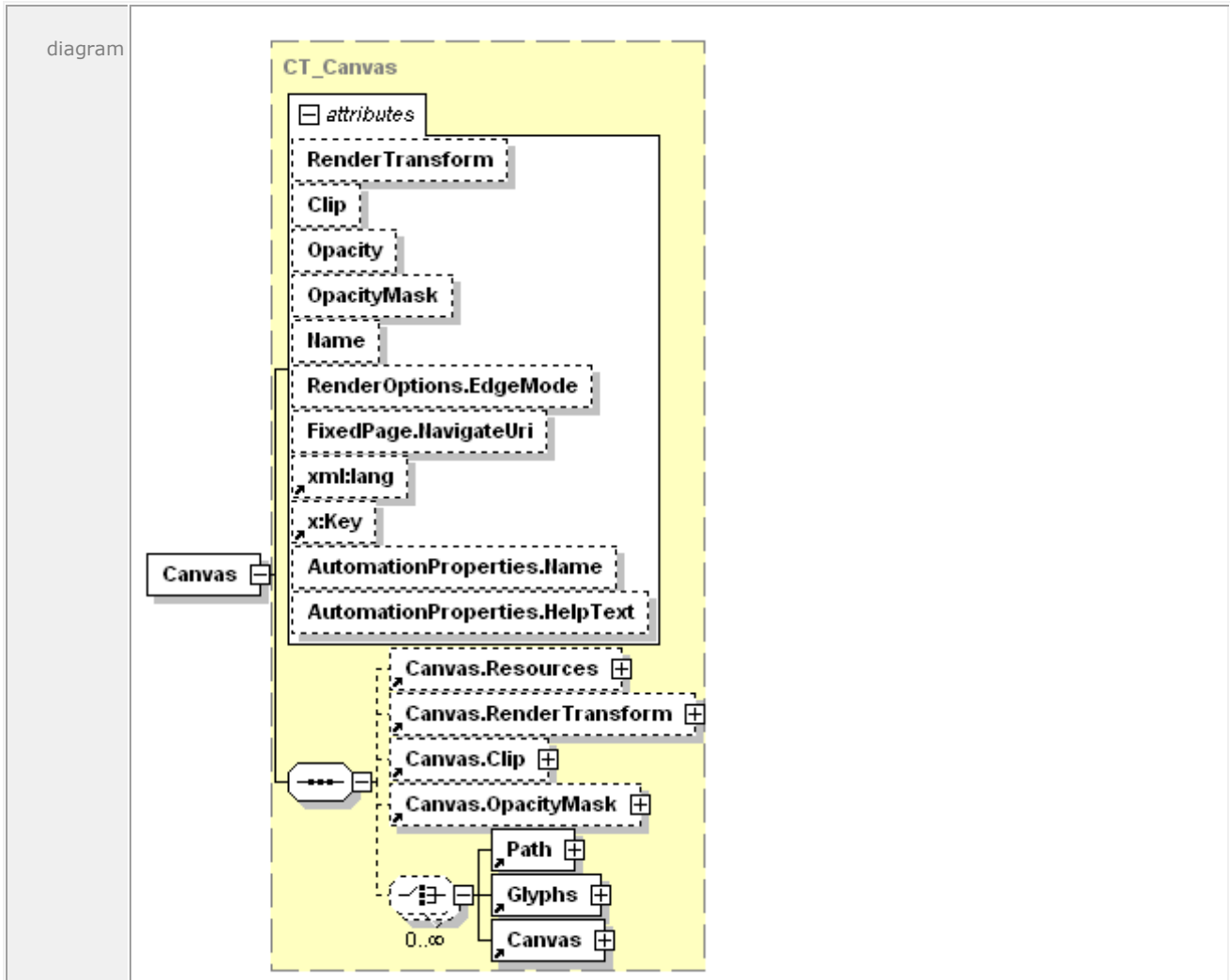
### 24 **10.3.3 Media Orientation and Scaling**

25 When rendering a FixedPage for printing, consumers are responsible for mapping from  
26 FixedPage content to the physical media. Differences in device capabilities and device  
27 configuration result in a large number of permutations for the mapping. The positioning,  
28 scaling, orientation, and clipping of FixedPage content when mapping to physical media MAY be  
29 controlled by settings provided in the PrintTicket [O3.1]. In the absence of settings provided in  
30 the PrintTicket, the mapping of FixedPage content to the physical media is implementation-  
31 defined.

32 By default, consumers SHOULD clip to the FixedPage Width and Height [S3.5]; consumers MAY  
33 provide implementation-defined mechanisms to select alternative clipping strategies [O3.2].  
34 [Note: For example, an implementation can provide a PrintTicket setting to allow control of  
35 consumer clipping of FixedPage content to one of the defined bounding boxes. *end note*]

1 **10.4 <Canvas> Element**

2 element **Canvas**



attributes	Name	Type	Use	Default	Fixed	Annotation
	RenderTransform	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				Limits the rendered region of the element.
	Opacity	<a href="#">ST_ZeroOne</a>		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 OpenXPS Document consumers. For more information, see §12.1.

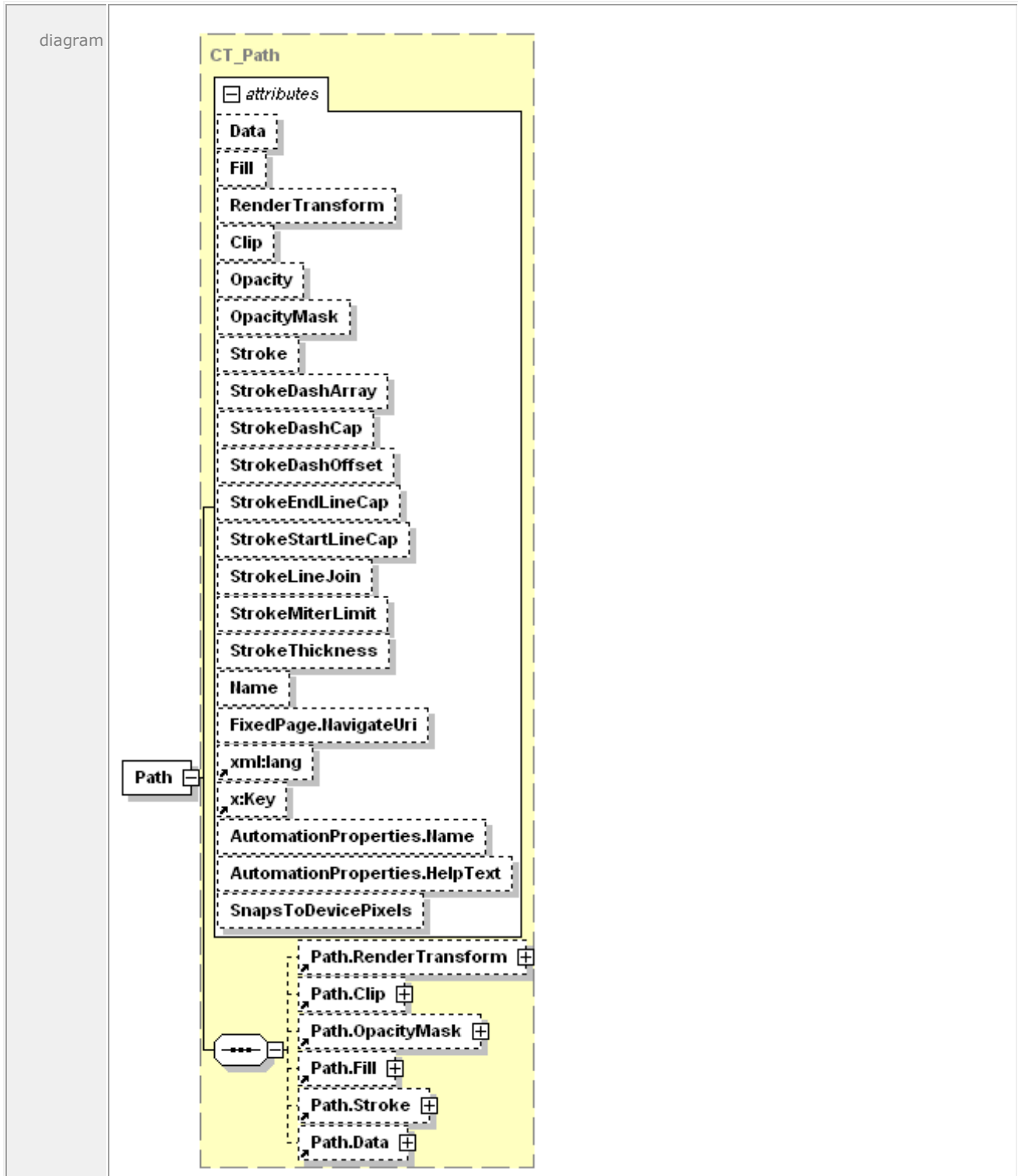
5

## 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>.  
8 3D graphics content is included in fixed-page markup by specifying a <Path> element filled  
9 with a <Brush3D>. The <Brush3D> element usage with Markup Compatibility is defined in  
10 Annex A.

# 1 11.1 <Path> Element

2 element **Path**





attributes	Name	Type	Use	Default	Fixed	Annotation
	Data	<a href="#">ST_RscRefAbbrGeomF</a>				Describes the geometry of the path.
	Fill	<a href="#">ST_RscRefColor</a>				Describes the brush used to paint the geometry specified by the Data property of the path.
	RenderTransform	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				Limits the rendered region of the element.
	Opacity	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRef</a>				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	<a href="#">ST_RscRefColor</a>				Specifies the brush used to draw the stroke.
	StrokeDashArray	<a href="#">ST_EvenArrayPos</a>				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, 3D graphics (Annex A), 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.

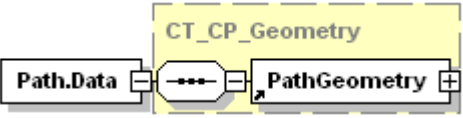
22 Consumers or viewers that perform anti-aliasing MAY “snap” those control points of the path  
 23 that are situated on the path bounding box to whole device pixels if the ignorable  
 24 SnapsToDevicePixels attribute is specified as true [O4.1].

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	 <pre> classDiagram     class PathData     class PathGeometry     class CT_CP_Geometry     PathData --&gt; PathGeometry     PathGeometry --&gt; CT_CP_Geometry           </pre>
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*

```

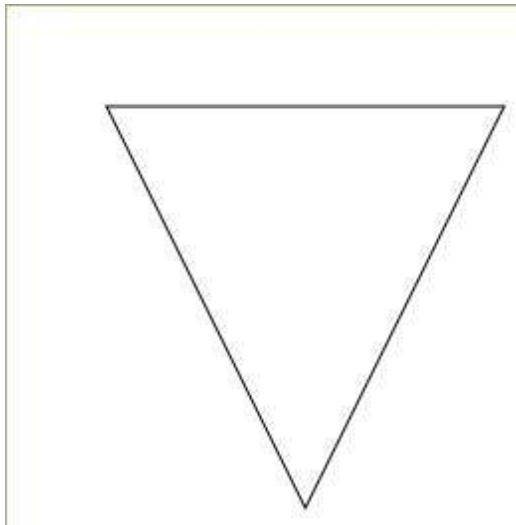
34     <Path Stroke="#000000" StrokeThickness="1">
35         <Path.Data>
36             <PathGeometry>
  
```

```

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

```

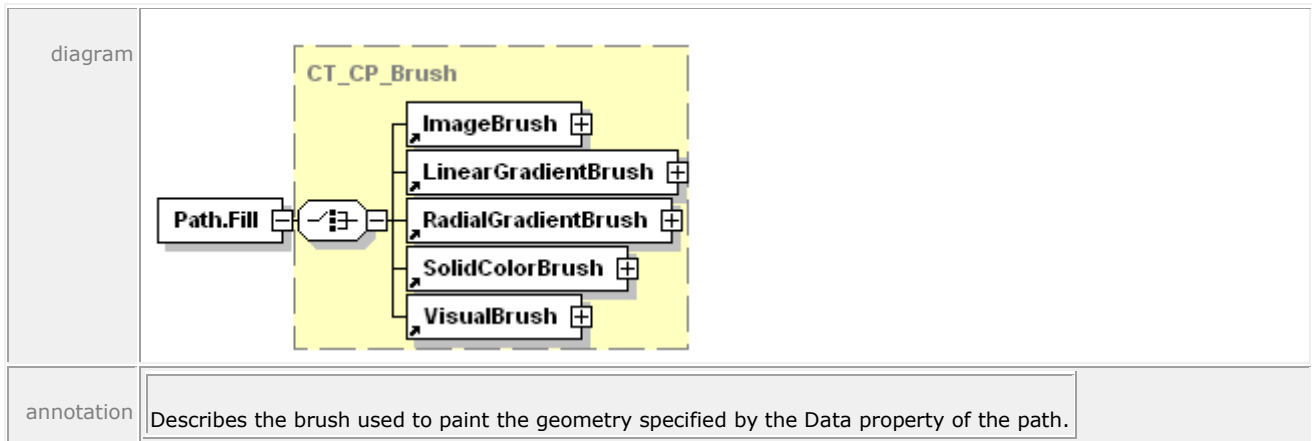
7 This markup produces the following results:



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**

diagram	<pre> classDiagram     class PathStroke     class CT_CP_Brush {         ImageBrush         LinearGradientBrush         RadialGradientBrush         SolidColorBrush         VisualBrush     }     PathStroke --&gt; CT_CP_Brush </pre>
annotation	Specifies the brush used to draw the 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).

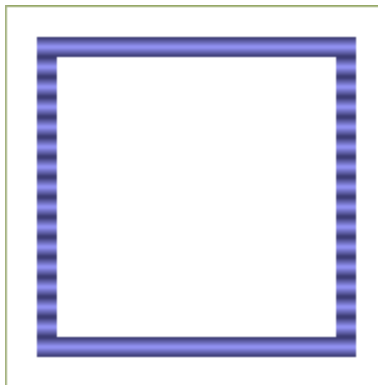
9 For more information, see §18.6.

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 [O4.2].

- 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	<a href="#">ST_AbbrGeom</a>				Describes the geometry of the path.
	FillRule	<a href="#">ST_FillRule</a>		EvenOdd		Specifies how the intersecting areas of geometric shapes are combined to form a region. Valid values are EvenOdd and NonZero.
	Transform	<a href="#">ST_RscRefMatrix</a>				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.

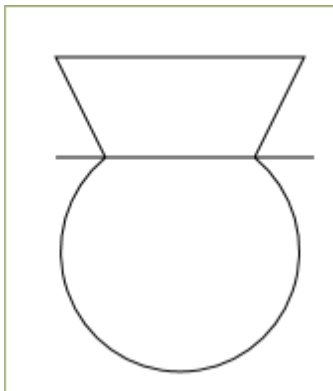
1 *Example 11-4. <PathGeometry> usage*

```

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]*

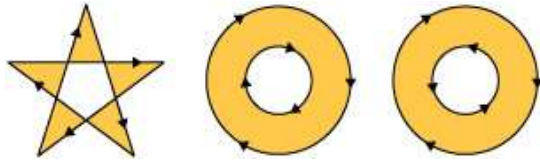
#### 25 **11.2.1.2 FillRule Attribute**

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 OpenXPS Document markup.

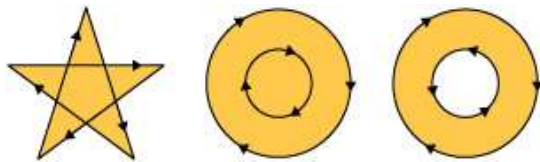
1 *Figure 11-1. Fill using EvenOdd algorithm*



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*



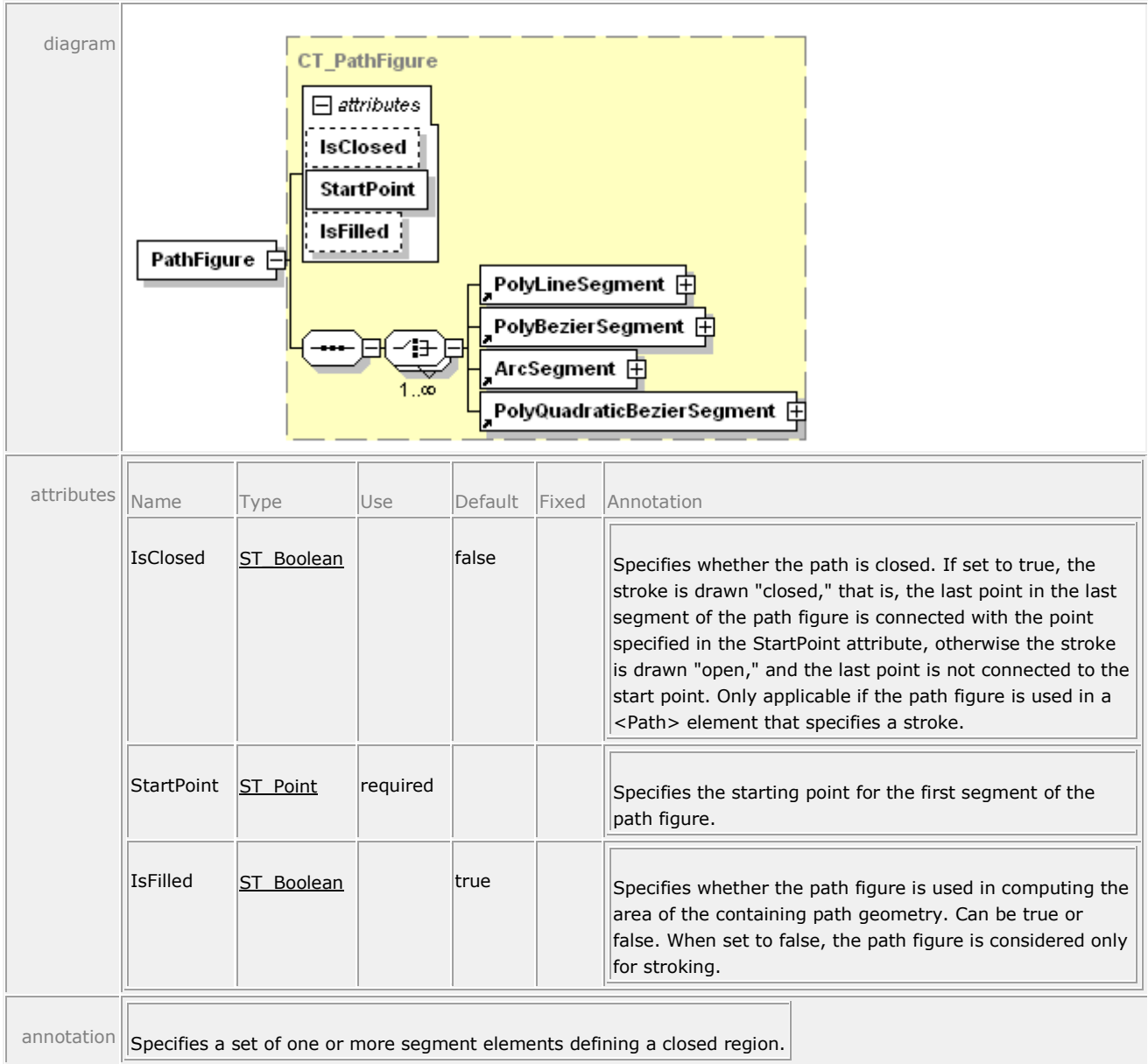
12 **11.2.1.3 Figures Attribute**

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 [M4.4].

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.

9 **11.2.2.2 <ArcSegment> Element**

10 element **ArcSegment**

diagram						
attributes	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.

annotation	Represents an elliptical arc between two points.
------------	--

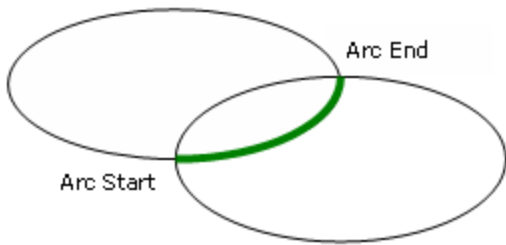
1 The <ArcSegment> element describes an elliptical arc. It is geometrically defined by the  
 2 intersection of two ellipses that have the same x radius and y radius. The ellipses intersect at  
 3 the starting and ending points of the arc.

4 *Table 11-1. Arc segment definition*

<b>Term</b>	<b>Description</b>
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. [ <i>Example: "100,50" end example</i> ]
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 $\geq 180^\circ$ ), and when it is false it uses the smaller arcs (arc length $< 180^\circ$ ).
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*

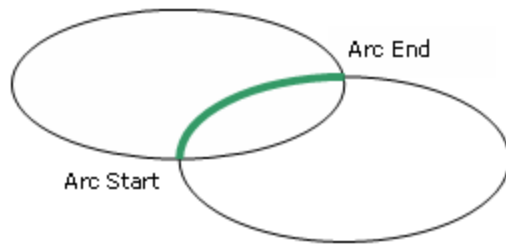
2 `IsLargeArc = false; SweepDirection = Counterclockwise`



3

4 *Figure 11-4. Arc choice B*

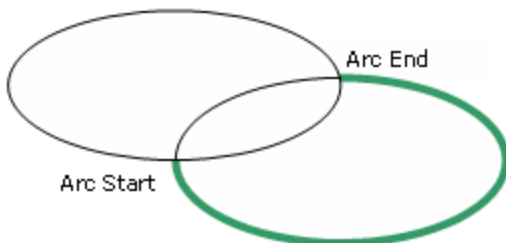
5 `IsLargeArc = false; SweepDirection = Clockwise`



6

7 *Figure 11-5. Arc choice C*

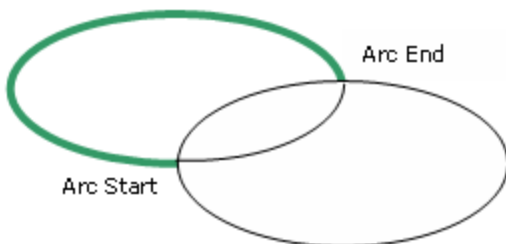
8 `IsLargeArc = true; SweepDirection = Counterclockwise`



9

10 *Figure 11-6. Arc choice D*

11 `IsLargeArc = true; SweepDirection = Clockwise`



12

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

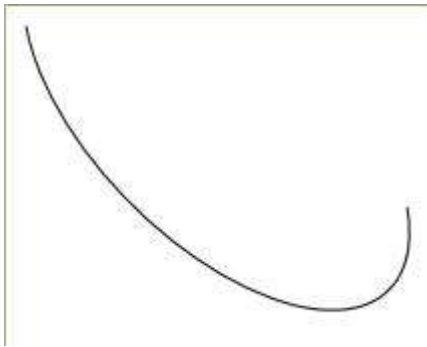
```
14 <Path Stroke="#000000" StrokeThickness="1">
15 <Path.Data>
```

```

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 [M4.5].
- 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**

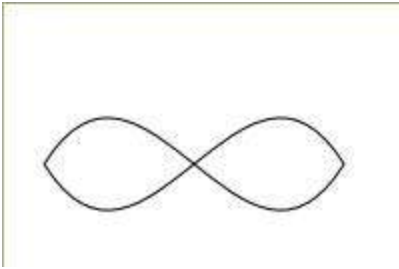
diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<a href="#">ST_PointsM3</a>	required			Specifies control points for multiple Bézier segments. Coordinate values within each pair are comma-separated and additional whitespace can appear. Coordinate pairs are separated from other coordinate pairs by whitespace.
	IsStroked	<a href="#">ST_Boolean</a>		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>

```



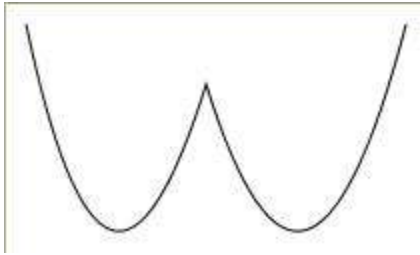
1 *Example 11-8. <PolyQuadraticBezierSegment> usage*

```

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*

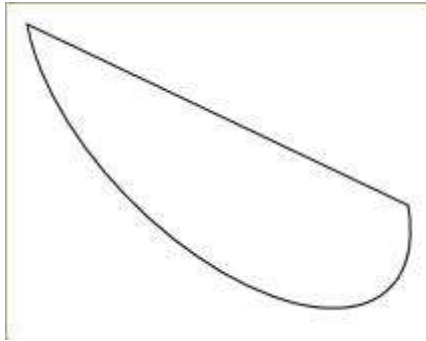
22 The following markup shows the arc segment as shown in Example 11-5 with the `IsClosed`  
 23 attribute of the `<PathFigure>` element set to `true`.

```

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 [O4.3]. 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 [one](#)  
9 [or more](#) drawing commands to create segments, and an optional Close command to create a  
10 closing segment. [The behavior of a degenerate geometry with no drawing commands is](#)  
11 [implementation-defined.](#) Drawing commands include:

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

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

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

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

28 If entering more than one drawing command of the same type sequentially, the duplicate  
29 command entry MAY be omitted [O4.4]. [*Example*: "L 100,200 300,400" is equivalent to "L  
30 100,200 L 300,400". *end example*] The current endpoint is determined as though each  
31 command appeared individually.

32 Values specifying coordinates can be real numbers.

1 For more information, see §B.

2 *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 <b>MUST</b> appear only as the first command in the abbreviated geometry syntax [M4.6]. This command <b>MUST NOT</b> be specified in the value of the Figures attribute of the <PathGeometry> element [M4.7]. [ <i>Example: F 0 end example</i> ]	<PathGeometry> FillRule attribute
Move	M x,y or m x,y	Establishes a new current endpoint. Every geometry <b>MAY</b> specify one or more figures, and <b>MAY</b> be preceded by a FillRule command where allowed [O4.5]. The first figure in a geometry <b>MUST</b> begin with a Move command [M4.8]. Subsequent Move commands indicate the start of a new figure but <b>MAY</b> be omitted, indicating the current endpoint for the subsequent figure is the same as the end point of the previous figure [O4.6]. [ <i>Example: M 1.0,1.5 end example</i> ]	<PathFigure> StartPoint attribute
Line	L x,y or l x,y	Draws a straight line from the current point to the specified point. [ <i>Example: L 20,30 end example</i> ]	<PolyLineSegment> element
Horizontal Line	H x or h x	Draws a horizontal line from the current endpoint to the specified x coordinate. [ <i>Example: H 90 end example</i> ]	<PolyLineSegment> element
Vertical Line	V y or v y	Draws a vertical line from the current endpoint to the	<PolyLineSegment> element

Name	Syntax	Description	Non-Abbreviated Equivalent
	<code>v y</code>	specified <i>y</i> coordinate. [ <i>Example: v 90 end example</i> ]	
Cubic Bézier Curve	<code>C x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub> x<sub>3</sub>, y<sub>3</sub></code> or <code>c x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub> x<sub>3</sub>, y<sub>3</sub></code>	Draws a cubic Bézier curve from the current endpoint to the specified point ( <i>x<sub>3</sub>, y<sub>3</sub></i> ) using the two specified control points ( <i>x<sub>1</sub>, y<sub>1</sub></i> and <i>x<sub>2</sub>, y<sub>2</sub></i> ). 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	<code>Q x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub></code> or <code>q x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub></code>	Draws a quadratic Bézier curve from the current endpoint to the specified point ( <i>x<sub>2</sub>, y<sub>2</sub></i> ) using the specified control point ( <i>x<sub>1</sub>, y<sub>1</sub></i> ). [ <i>Example: q 100, 200 300, 200 end example</i> ]	<PolyQuadraticBezierSegment> element
Smooth Cubic Bézier Curve	<code>S x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub></code> or <code>s x<sub>1</sub>, y<sub>1</sub> x<sub>2</sub>, y<sub>2</sub></code>	Draws a cubic Bézier curve from the current endpoint to the specified point ( <i>x<sub>2</sub>, y<sub>2</sub></i> ). 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 <i>x<sub>1</sub>, y<sub>1</sub></i> . [ <i>Example: S 100, 200 200, 300 end example</i> ]	<PolyBezierSegment> element
Elliptical Arc	<code>A x<sub>r</sub>, y<sub>r</sub> r<sub>x</sub> fArc fSweep</code> <i>x, y</i> or <code>a x<sub>r</sub>, y<sub>r</sub> r<sub>x</sub> fArc fSweep</code> <i>x, y</i>	Draws an elliptical arc from the current endpoint to the specified point ( <i>x, y</i> ). The size and orientation of the ellipse are defined by <i>x<sub>r</sub>, y<sub>r</sub></i> , <i>r<sub>x</sub>, r<sub>y</sub></i> defines the <i>x</i> radius, <i>y<sub>r</sub></i> defines the <i>y</i> radius, and <i>r<sub>x</sub></i> defines the <i>x</i> -axis rotation in degrees,	<ArcSegment> element

Name	Syntax	Description	Non-Abbreviated Equivalent
		<p>which indicates how the 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. <math>f_{Arc}</math> and <math>f_{Sweep}</math> 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 <math>f_{Arc}</math> is 1, one of the two larger arc sweeps is chosen. If <math>f_{Arc}</math> is 0, one of the smaller arc sweeps is chosen. No other values of <math>f_{Arc}</math> are valid.</p> <p>If <math>f_{Sweep}</math> is 1, the arc is drawn in a positive-angle (clockwise) direction. If <math>f_{Sweep}</math> is 0, the arc is drawn in a negative-angle (counter-clockwise) direction. No other values of <math>f_{Sweep}</math> 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:



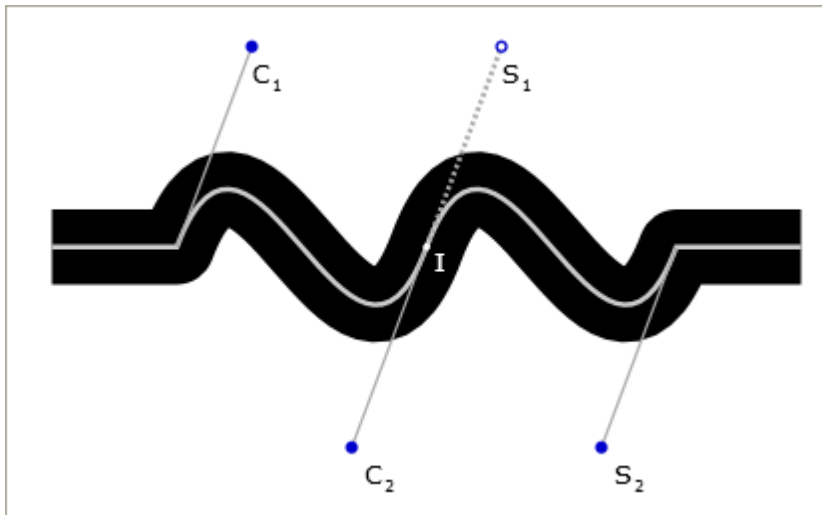
1       <Path Stroke="#000000" Data="M 100,100 L 300,100 L 200,300 z" />  
 2 *end example]*

### 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]*

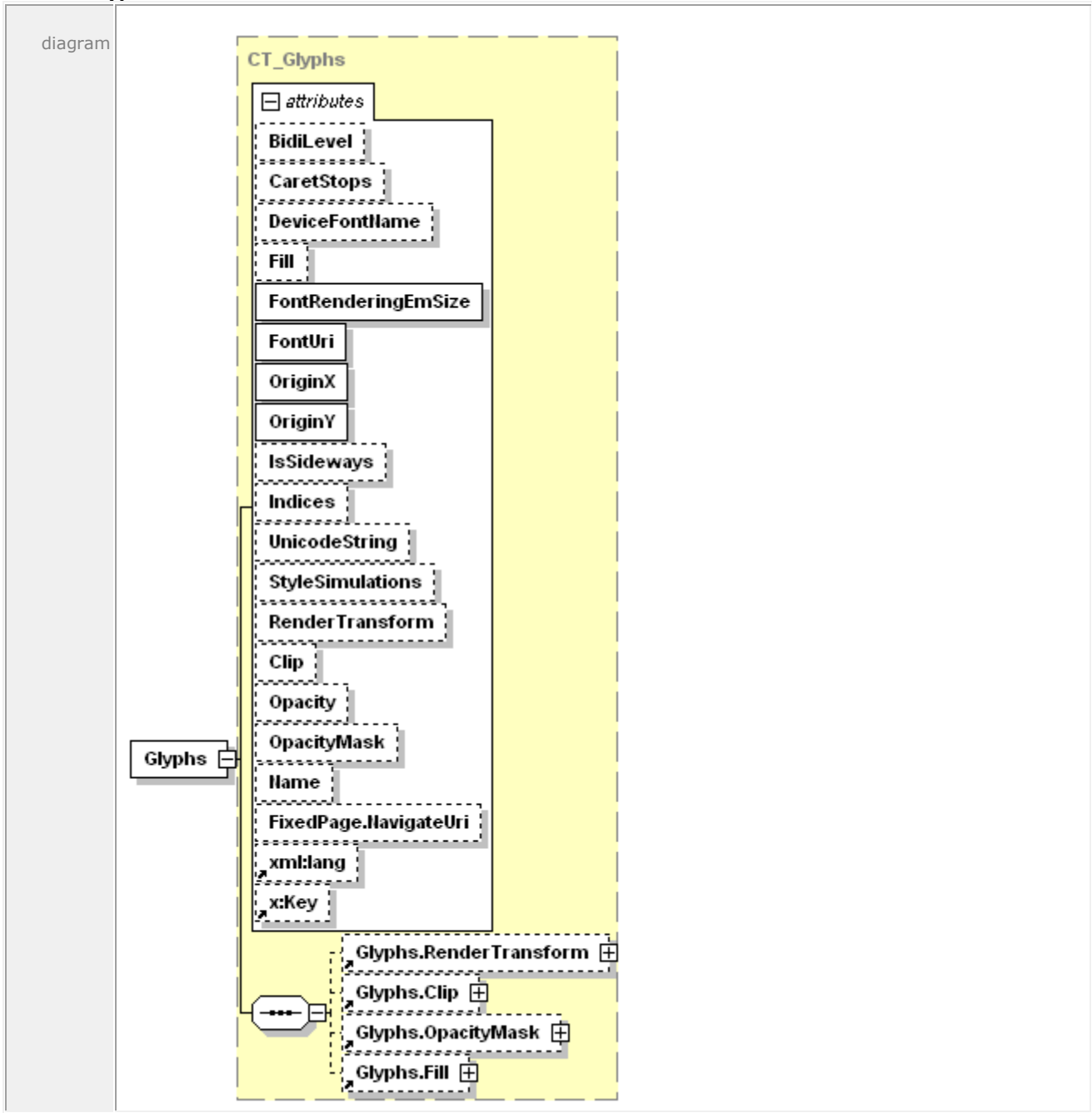


## 1 **12. Text**

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. <del>If the UnicodeString attribute specifies an empty string (" or ") and the Indices attribute is not specified or is also empty, a consumer MUST instantiate an error condition.</del> <a href="#">If the UnicodeString attribute of the &lt;Glyphs&gt; element is not specified or contains an empty value (" or "{ }"), and if the Indices attribute is not specified or contains &lt;nothing&gt;, then a consumer MUST instantiate an error condition.</a> [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. <a href="#">If the UnicodeString attribute of the &lt;Glyphs&gt; element is not specified or contains an empty value ("</a>



					<a href="#">or "{ }"), and if the Indices attribute is not specified or contains &lt;nothing&gt;, then a consumer MUST instantiate an error condition [M5.2].</a>
StyleSimulations	<a href="#">ST_StyleSimulations</a>		None		Specifies a style simulation. Valid values are None, ItalicSimulation, BoldSimulation, and BoldItalicSimulation.
RenderTransform	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				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	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRef</a>				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	<a href="#">ST_Name</a>				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

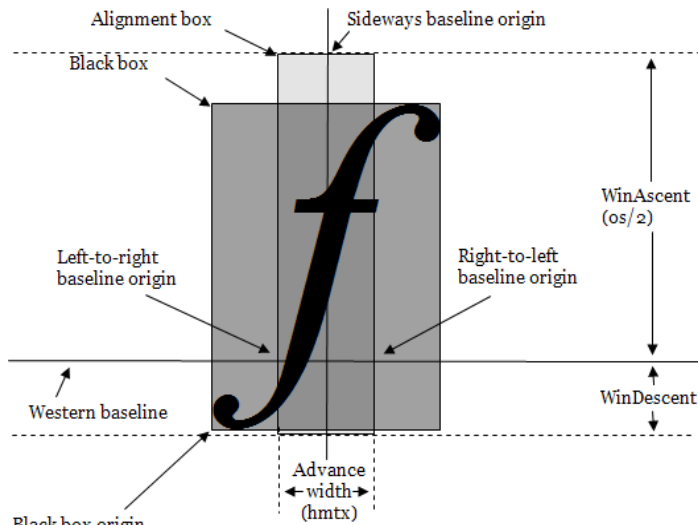
						resource that is internal to or external to the package.
	xml:lang					Specifies the default language used for the current element. The language is specified according to RFC 3066.
	x:Key					Specifies a name for a resource in a resource dictionary. x:Key MUST be present when the current element is defined 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**

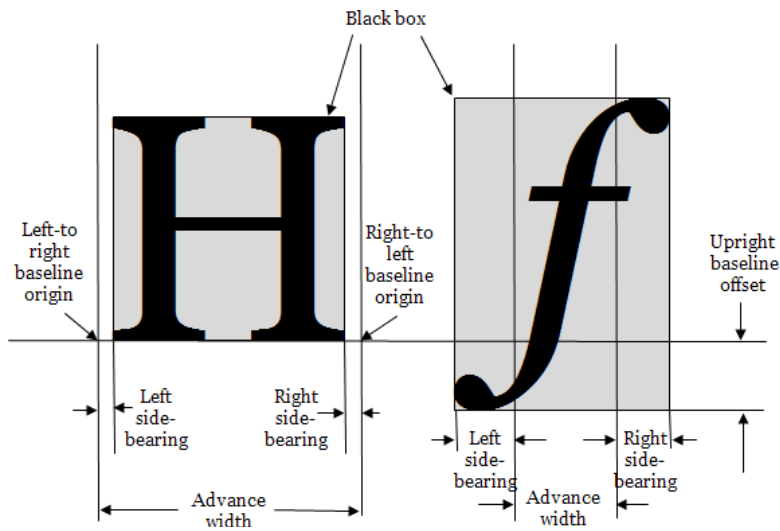
2 Each glyph defines metrics that specify how it aligns with other glyphs. The metrics are  
 3 illustrated below.

4 *Figure 12-1. Glyph metrics*[GG2]

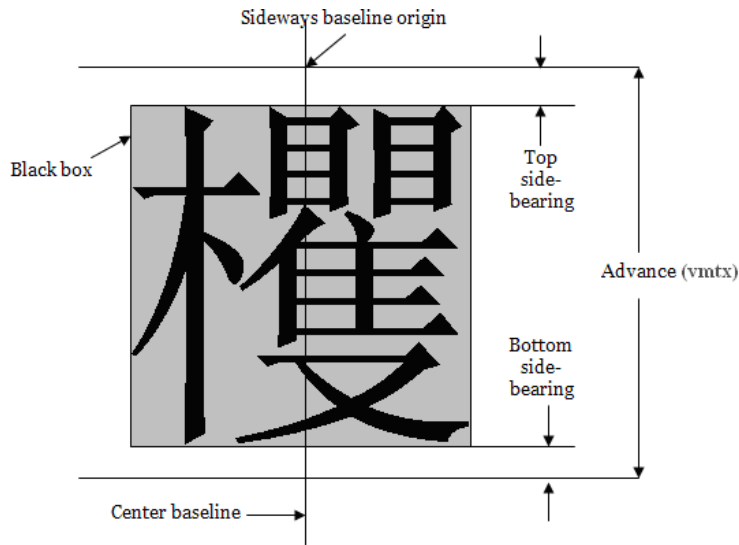


5 Black box origin

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 glyph  
 5 offset vector. Combining marks usually have a zero advance width. The glyph offset vector can  
 6 be used to adjust the position of a combining mark and, therefore, can have a non-0,0 value for  
 7 combining 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 (as set by `uOffset` and `vOffset` in the  
 15 Indices attribute; see §12.1.3) adjusts the position of this glyph relative to its nominal  
 16 origin. The orientation of the glyph offset vector is not affected by the value of the  
 17 `lsSideways` attribute, but is affected by the value of the `BidiLevel` 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.

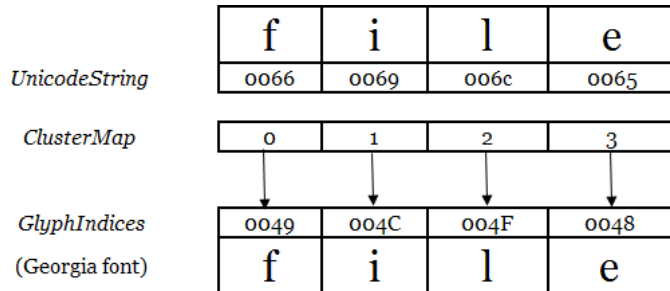
26 The cluster map contains one entry for each UTF-16 code unit in the `UnicodeString` attribute.  
 27 Each entry specifies the offset of the first glyph that represents the cluster of UTF-16 code  
 28 units.

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.



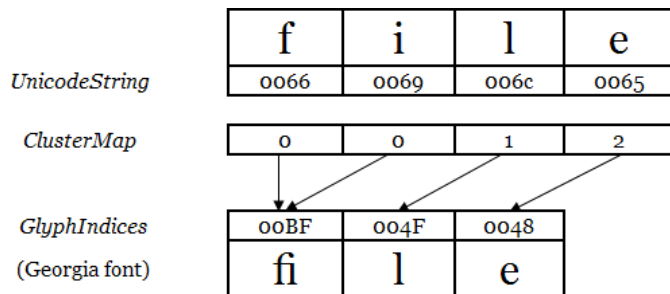
6  
7 *end example]*

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.



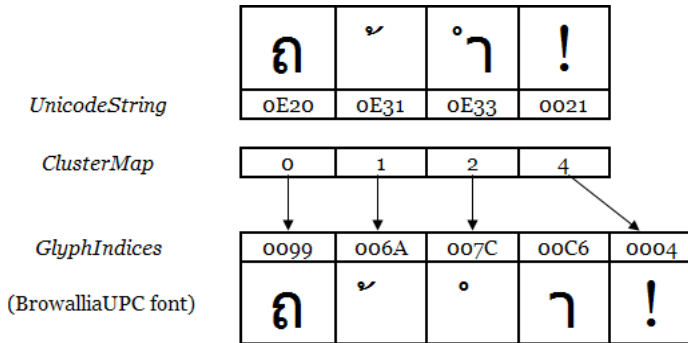
13  
14 *end example]*

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.



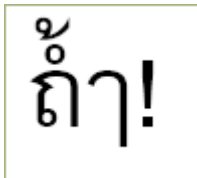
10

11 The markup appears as follows:

```

12 <Glyphs
13   FontUri="../Resources/Fonts/browau.ttf"
14   UnicodeString="&#xe20;&#xe31;&#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]*

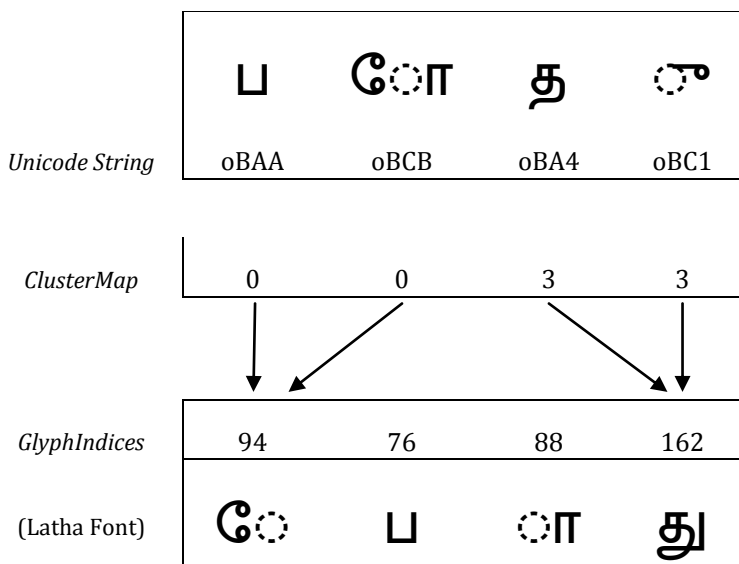
#### 12.1.2.4 Many-to-Many Mappings

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

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

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

12



13

The markup appears as follows:

```

15 <Glyphs
16   FontUri="../Resources/Fonts/latha.ttf"
17   UnicodeString="#&#xbaa;&#x bcb;&#x ba4;&#x bc1;"
18   Indices="(2:3)94;76;88;(2:1)162"
19   OriginX="10" OriginY="120"
20   FontRenderingEmSize="40"
21   Fill="#000000"/>

```

The markup above is rendered as follows:

23

1 *end example]*

### 2 **12.1.3 Indices Attribute**

3 The <Glyphs> element MAY have an Indices attribute [O5.7]. The glyph specifications within the  
4 Indices attribute are OPTIONAL [O5.8]. The GlyphIndex portion of the Indices attribute MAY be  
5 used to specify a series of glyphs, complex character-to-glyph cluster mappings, or a  
6 combination of both [O5.9]. The Indices attribute MAY also include glyph placement information  
7 [O5.10].

8 Within the Indices attribute, each glyph specification is separated by a semicolon. The Indices  
9 attribute MUST adhere to the glyph specification syntax as follows [M5.25]:

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

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

29 *Table 12-3. Glyph specifications*

Name	Description
GlyphIndex	<p>Index of the glyph (16-bit) in the physical font. The entry MAY be empty [O5.11], 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>



---

Advance width is a real number with units specified in hundredths of 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 [M5.25]. The right-to-left writing direction can be specified using the BidiLevel attribute.

---

uOffset, vOffset    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.

Measured in hundredths of the font em size. The default offset values are 0.0,0.0. uOffset and vOffset are real numbers.

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.

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.

---

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    [O5.12].

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*

---

<b>Name</b>	<b>Description</b>
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 of the <Glyphs> element is not specified or contains an empty  
20 value (“” or “{”), and if the Indices attribute is not specified or contains <nothing>, then a  
21 consumer MUST instantiate an error condition ~~If the UnicodeString attribute specifies an empty~~  
22 ~~string (“” or “{”), and the Indices attribute is missing or is also empty, a consumer MUST~~  
23 ~~instantiate an error condition~~ [M5.2]. If the UnicodeString attribute contains a Unicode code unit  
24 that cannot be mapped to a glyph index via a cmap table in the font and there is no  
25 corresponding GlyphIndex entry in the Indices attribute, the consumer MUST display the .notdef  
26 glyph [M5.9].

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

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

### 12.1.5 StyleSimulations Attribute

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

When the StyleSimulations value is specified as BoldSimulation, synthetic emboldening is applied by geometrically widening the strokes of glyphs by  $\pm 1\%$  of the em size [for each of the two boundaries of the stroke](#), so that the centers of strokes remain at the same position [relative to the character coordinate system](#). This leaves the baseline origin unmodified. The black box grows 1% all around for a total of 2% horizontal and 2% vertical. As a result, the character height and the advance width of each glyph are increased by 2% of the em size. Producers MUST lay out algorithmically emboldened glyphs using advance widths that are 2% of the em size larger than when not algorithmically emboldened [M5.12]. [When rendering glyphs where the StyleSimulations value is specified as BoldSimulation, consumers SHOULD offset each glyph up and to the right by 1% of the em size so that baseline and left edge alignments are preserved \[S5.6\]](#).

Consumers MUST implement the effect of algorithmic emboldening such that the black box of the glyph grows by 2% of the em size [M5.13]. 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 [M5.13].

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

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

### 12.1.6 IsSideways Attribute

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

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

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

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

#### 6 **12.1.6.1 Calculating Sideways Text Origin and Advance Width**

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

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

13 
$$\text{TopOriginX} = \text{hmtx.advanceWidth}[\text{GlyphIndex}] / 2$$

14 If the font is a CFF Open [Font Format Type](#) font, the y origin of the unturned glyph is  
 15 determined from the vertical origin (`vorg`) table for the font, which can be specified for a  
 16 particular glyph index but falls back to the default vertical origin if the glyph index is not  
 17 present in the vertical origin table. This formula is expressed as follows (in pseudocode):

18 
$$\text{TopOriginY} = \text{vorg.vertOriginY}[\text{glyphIndex}]$$

19 or:

20 
$$\text{TopOriginY} = \text{vorg.defaultVertOriginY}$$

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

25 
$$\text{TopOriginY} = \text{glyf.yMax}[\text{glyphIndex}] + \text{vmtx.topSideBearing}[\text{glyphIndex}]$$

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

29 If the vertical metrics font table does not exist but the "OS/2" metrics table does exist and is at  
 30 least 78 bytes long, the "OS/2" table is consulted and the `sTypoAscender` and `sTypoDescender`  
 31 values are used, as follows (in pseudocode):

32 
$$\text{TopOriginY} = \text{os/2.sTypoAscender}$$
  
 33 
$$\text{Descender} = \text{abs}(\text{os/2.typoDescender})$$

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

36 
$$\text{TopOriginY} = \text{hhea.Ascender}$$
  
 37 
$$\text{Descender} = \text{abs}(\text{hhea.Descender})$$

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

1        AdvanceWidth = TopOriginY + Descender

### 2    **12.1.6.2 IsSideways and BidiLevel Effects on Glyph Positioning**

3    Right-to-left text (BidiLevel attribute set to an odd value) changes the direction of the  
4    AdvanceWidth and uOffset (horizontal offset) values of the Indices attribute, as well as the  
5    position of the glyph origin. Vertical text (IsSideways attribute set to true) changes the position  
6    of the glyph origin.

7    Producers MUST NOT specify text that is both right-to-left (BidiLevel attribute set to an odd  
8    value) and vertical (IsSideways attribute set to true) [M5.15].

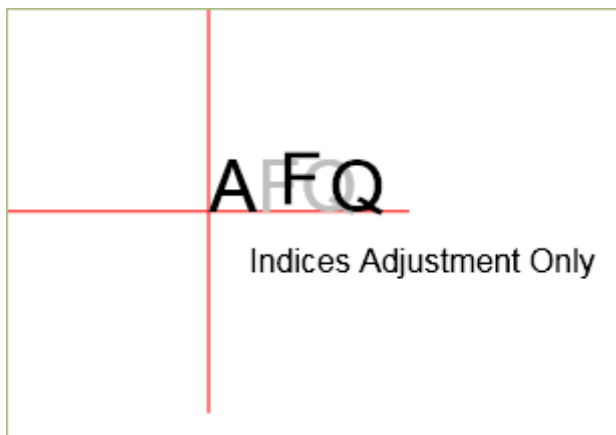
9    *Table 12-5. IsSideways and BidiLevel effects on origin placement*

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

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

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

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



19

1 *end example]*

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

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

```
5     <Glyphs Fill="#000000" FontRenderingEmSize="48"
6         OriginX="100" OriginY="100"
7         UnicodeString="AFQ"
8         Indices=";,100,30,10;"
9         BidiLevel="1"
10        FontUri=" ../Resources/Fonts/Arial.ttf" />
```



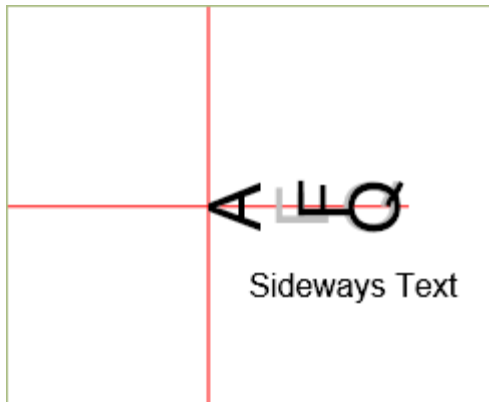
11

12 *end example]*

13 *Example 12–9. Sideways text (IsSideways set to true)*

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

```
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" />
```

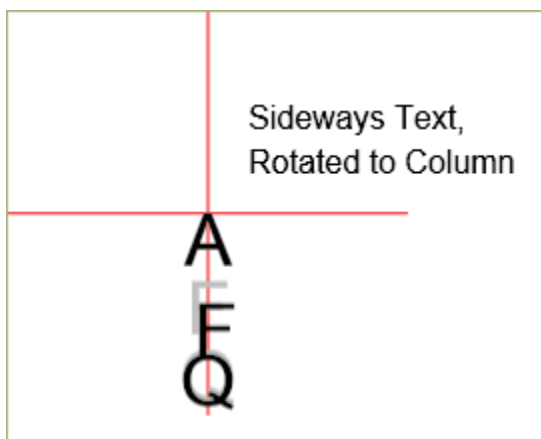


1  
2 *end example]*

3 *Example 12-10. Vertical text*

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

```
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        RenderTransform="0,1,-1,0,200,0" />
```



14  
15 *end example]*

16 *Example 12-11. Japanese vertical text*

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

```

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

```

17 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。

18

19 *end example]*



### 12.1.7 DeviceFontName Attribute

Printer device fonts are specified by the DeviceFontName attribute. Device manufacturers define the values for this attribute. 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 [S5.4].

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

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

For producers, a <Glyphs> element with a specified device font name MUST have exactly one Indices glyph per code unit 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 and MUST render the embedded version of the font [M5.19]. This means that a device font cannot be used for characters outside the basic multilingual plane (BMP).

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

### 12.1.8 xml:lang Attribute

OpenXPS Document consumers might need to override the default language for a specific run of glyphs, particularly in multilingual documents. The language defaults to the value specified for the xml:lang attribute of the <FixedPage> element but MAY be overridden by an xml:lang attribute on a <Glyphs> element [O5.13]. For larger blocks of text, the producer MAY specify the xml:lang attribute on the <Canvas> element [M5.27].

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

### 12.1.9 CaretStops Attribute

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

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

If the CaretStops attribute is omitted, it is legal to place the caret before any of the UTF-16 code units in the Unicode string. Therefore, omitting the CaretStops attribute is equivalent to

1 specifying a string that has all the bits set to 1. If there are insufficient flags in the CaretStops  
 2 string to correspond to all the UTF-16 code units in the Unicode string, all remaining UTF-16  
 3 code units in the Unicode string MUST be considered valid caret stops [M5.22].

4 *Example 12–12. Using the CaretStops attribute to determine a valid caret stop position*

5 Given the following attributes, the *m* in “example” is not a valid caret stop position:

```
6 UnicodeString = "This is an example string of text."  
7 CaretStops = "ffff"
```

8 *end example]*

### 9 **12.1.10 Optimizing Glyph Markup**

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

#### 13 **12.1.10.1 Optimizing Glyph Indices Markup**

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

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

#### 18 **12.1.10.2 Optimizing Glyph Position Markup**

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

- 20 • For glyphs that have not been algorithmically emboldened, the desired advance width is  
 21 the value listed in the horizontal metrics font table (if the IsSideways attribute value is  
 22 false) or as calculated in §12.1.6.1 (if the IsSideways attribute value is true).
- 23 • For algorithmically emboldened glyphs, the desired advance width is exactly 2% larger  
 24 than the values in the horizontal metrics font table (if the IsSideways attribute value is  
 25 false) or as calculated in §12.1.6.1 (if the IsSideways attribute value is true).

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

### 30 **12.1.11 Glyph Markup Examples**

31 *Example 12–13. Basic italic font*

```
32 <Canvas>  
33 <Glyphs  
34 FontUri="../Resources/Fonts/Timesi.ttf"  
35 FontRenderingEmSize="20"  
36 OriginX="35"  
37 OriginY="35"  
38 UnicodeString="Basic italic font..."  
39 Fill="#009900" />  
40 </Canvas>
```

1 This text is rendered as follows:



*Basic italic font...*

2

3 *end example]*

4 *Example 12-14. Italic font using StyleSimulations attribute*

```
5 <Canvas>
6 <Glyphs
7   FontUri=" ../Resources/Fonts/Times.ttf"
8   FontRenderingEmSize="20"
9   StyleSimulations="ItalicSimulation"
10  OriginX="35"
11  OriginY="35"
12  UnicodeString="Simulated italic font..."
13  Fill="#009900" />
14 </Canvas>
```

15 This text is rendered as follows:



*Simulated italic font...*

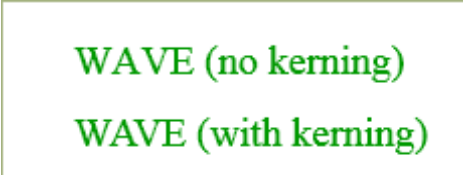
16

17 *end example]*

18 *Example 12-15. Kerning*

```
19 <Canvas>
20
21 <!-- "WAVE" without kerning -->
22
23 <Glyphs
24   OriginX="35"
25   OriginY="35"
26   UnicodeString="WAVE (no kerning)"
27   FontUri=" ../Resources/Fonts/Times.ttf"
28   FontRenderingEmSize="20"
29   Fill="#009900" />
30
31 <!-- "WAVE" with kerning -->
32
33 <Glyphs
34   OriginX="35"
35   OriginY="70"
36   UnicodeString="WAVE (with kerning)"
37   Indices=",88;,59"
38   FontUri=" ../Resources/Fonts/Times.ttf"
39   FontRenderingEmSize="20"
40   Fill="#009900" />
41
42 </Canvas>
```

1 This text is rendered as follows:



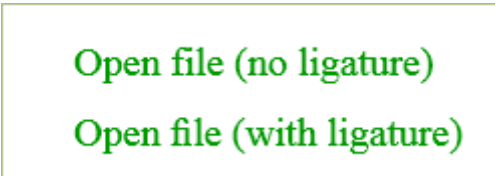
2

3 *end example]*

4 *Example 12-16. Ligatures*

```
5 <Canvas>
6
7 <!-- "Open file" without "fi" ligature -->
8
9 <Glyphs
10   OriginX="35"
11   OriginY="35"
12   UnicodeString="Open file (no ligature)"
13   FontUri=" ../Resources/Fonts/Times.ttf"
14   FontRenderingEmSize="20"
15   Fill="#009900" />
16
17 <!-- "Open file" with "fi" ligature -->
18
19 <Glyphs
20   OriginX="35"
21   OriginY="70"
22   UnicodeString="Open file (with ligature)"
23   Indices=";;;;(2:1)191"
24   FontUri=" ../Resources/Fonts/Times.ttf"
25   FontRenderingEmSize="20"
26   Fill="#009900" />
27
28 </Canvas>
```

29 This text is rendered as follows:



30

31 *end example]*

32 *Example 12-17. Cluster maps*

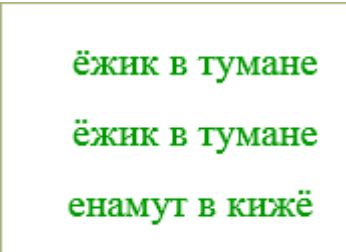
```
33 <Canvas>
34
35 <!-- "ёжик в тумане" using pre-composed "ё" -->
36
37 <Glyphs
38   OriginX="35"
39   OriginY="35"
```

```

1      xml:lang="ru-RU"
2      UnicodeString="ёжик в тумане"
3      FontUri=" ../Resources/Fonts/Times.ttf"
4      FontRenderingEmSize="20"
5      Fill="#009900" />
6
7      <!-- "ёжик в тумане" using composition of "e" and diaeresis -->
8
9      <Glyphs
10     OriginX="35"
11     OriginY="70"
12     xml:lang="ru-RU"
13     UnicodeString="ёжик в тумане"
14     Indices="(1:2)72;142,0,-40"
15     FontUri=" ../Resources/Fonts/Times.ttf"
16     FontRenderingEmSize="20"
17     Fill="#009900" />
18
19     <!-- "ёжик в тумане" Forced rendering right-to-left showing
20     combining mark in logical order -->
21
22     <Glyphs
23     OriginX="155"
24     OriginY="105"
25     BidiLevel="1"
26     xml:lang="ru-RU"
27     UnicodeString="ёжик в тумане"
28     Indices="(1:2)72;142,0,-40"
29     FontUri=" ../Resources/Fonts/Times.ttf"
30     FontRenderingEmSize="20"
31     Fill="#009900" />
32
33 </Canvas>

```

34 This text is rendered as follows:



ёжик в тумане  
ёжик в тумане  
енамут в кижё

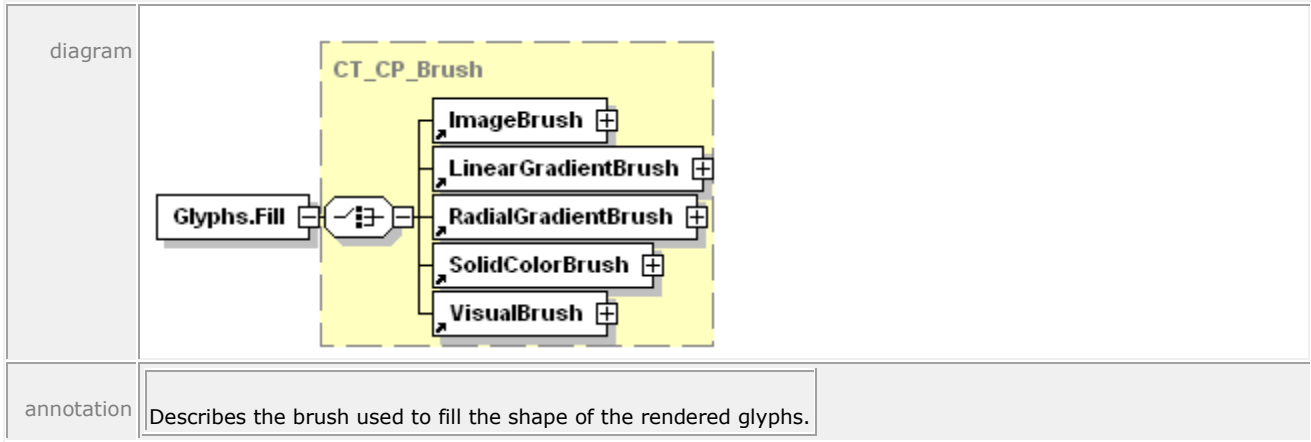
35

36 *end example*]

---

## 37 12.2 <Glyphs.Fill> Element

38 element **Glyphs.Fill**



- 1 The Fill property specifies the brush that fills a glyph. Any brush can be used.
- 2

## 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 Annex A defines an additional (optional) brush for the representation of 3D content.

15 *Table 13-1. Brush types*

<b>Name</b>	<b>Description</b>
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.
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.

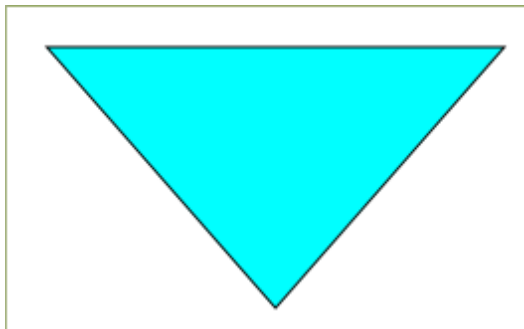


1 *Example 13-1. <SolidColorBrush> usage*

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:



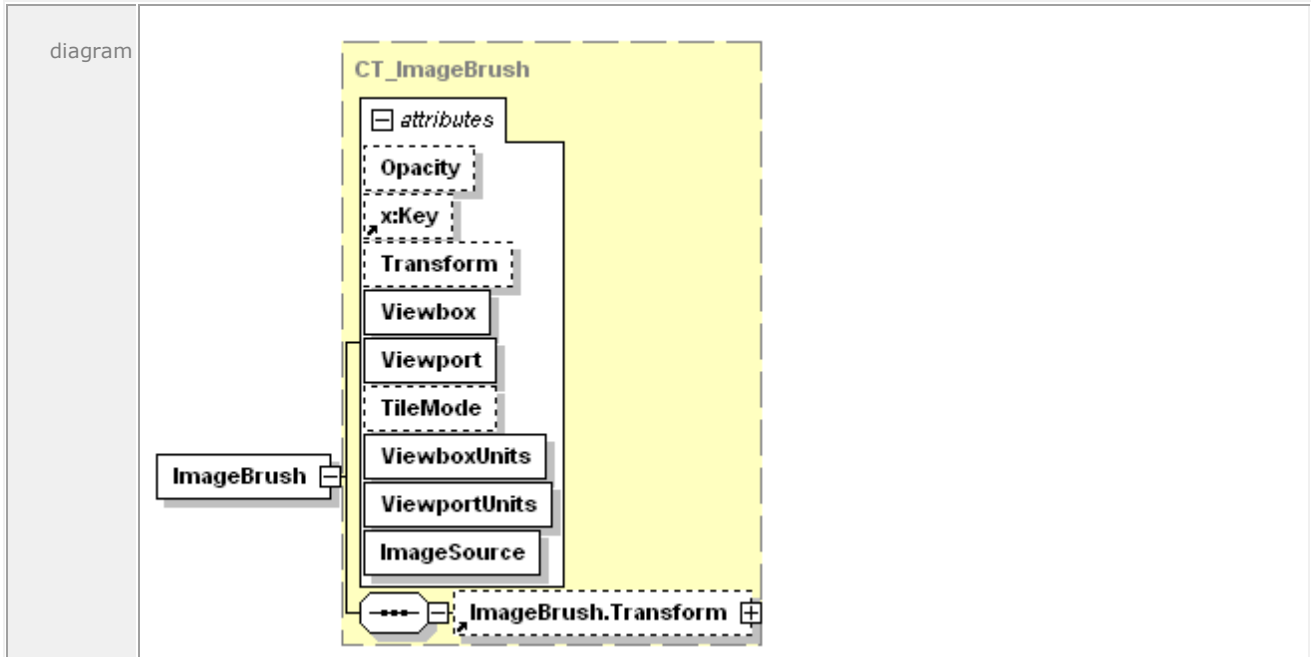
16

17 *end example]*

---

## 18 **13.2 <ImageBrush> Element**

19 element **ImageBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<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 [M6.2].
	Transform	<u>ST_RscRefMatrix</u>				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	<u>ST_ViewBox</u>	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	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_TileMode</a>		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
	ViewboxUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
	ViewportUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
	ImageSource	<a href="#">ST UriCtxBmp</a>	required			Specifies the URI of an image resource or a combination of the URI of an image resource a color profile resource. See §15.3.8. 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, or ~~Windows Media Photo~~[JPEG XR](#) image part within the OpenXPS Document package  
 4 [M6.3]. 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.

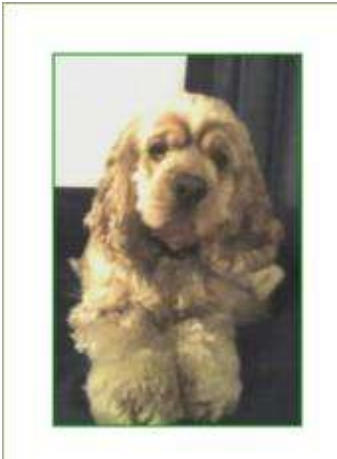
8 *Example 13–2. <ImageBrush> usage*

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"
19         ViewportUnits="Absolute" />
```

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

11 This markup produces the following results:

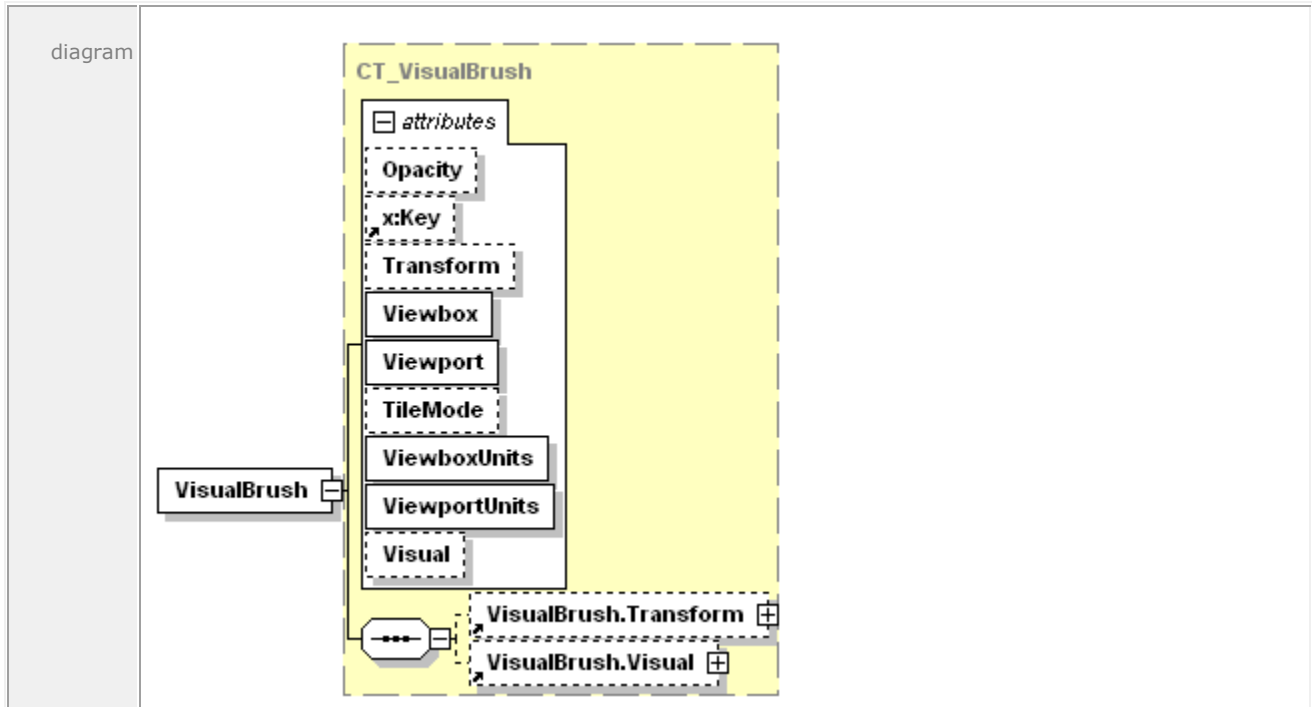


12

13 *end example]*

1 **13.3 <VisualBrush> Element**

2 element **VisualBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Opacity	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_ViewBox</a>	required			Specifies the position and dimensions of the brush's source content. Specifies four comma-separated real numbers (x, y, width, height),

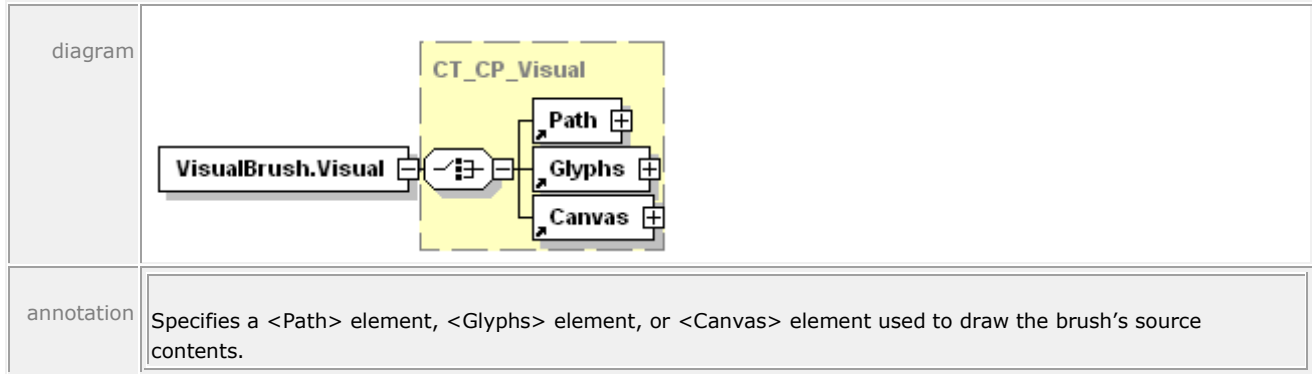
					where width and height are non-negative. The viewbox defines the default coordinate system for the element specified in the <VisualBrush.Visual> property element. The corners of the viewbox are mapped to the corners of the viewport, thereby providing the default clipping and transform for the brush's source content.
Viewport	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_TileMode</a>		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
Visual	<a href="#">ST_RscRef</a>				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.

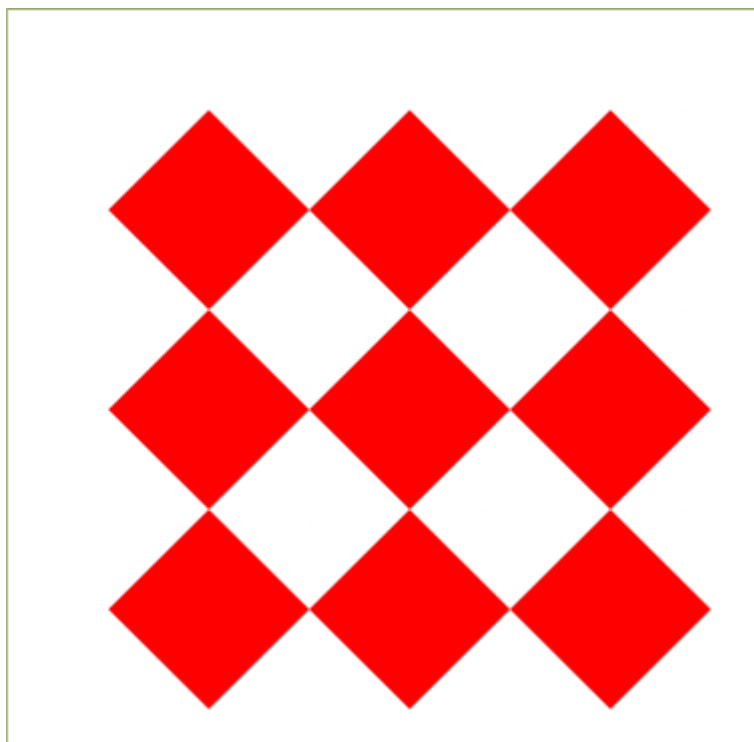
7 *Example 13–3. <VisualBrush.Visual> usage*

```

8     <Path>
9         <Path.Fill>
10            <VisualBrush
11                Viewbox="0,0,1,1"
12                Viewport="50,50,100,100"
13                ViewportUnits="Absolute"
14                ViewboxUnits="Absolute"
15                TileMode="Tile">
16                <VisualBrush.Visual>
17                    <Path>
18                        <Path.Fill>
19                            <SolidColorBrush Color="#FF0000" />
20                        </Path.Fill>
21                        <Path.Data>
22                            <PathGeometry>
23                                <PathFigure StartPoint="0,0.5" IsClosed="true">
24                                    <PolylineSegment Points="0.5,0 1.0,0.5
25                                        0.5,1.0" />
26                                </PathFigure>
27                            </PathGeometry>
28                        </Path.Data>
29                    </Path>
30                </VisualBrush.Visual>
31            </VisualBrush>
32        </Path.Fill>
33        <Path.Data>
34            <PathGeometry>
35                <PathFigure StartPoint="50,50" IsClosed="true">
36                    <PolylineSegment Points="350,50 350,350 50,350" />
37                </PathFigure>
38            </PathGeometry>
39        </Path.Data>
40    </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" [M6.7].
ViewportUnits	Specifies the unit type for the Viewport attribute. MUST have the value "Absolute" [M6.8].
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, where HorizontalImageResolution and VerticalImageResolution are specified in dpi:

```

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**

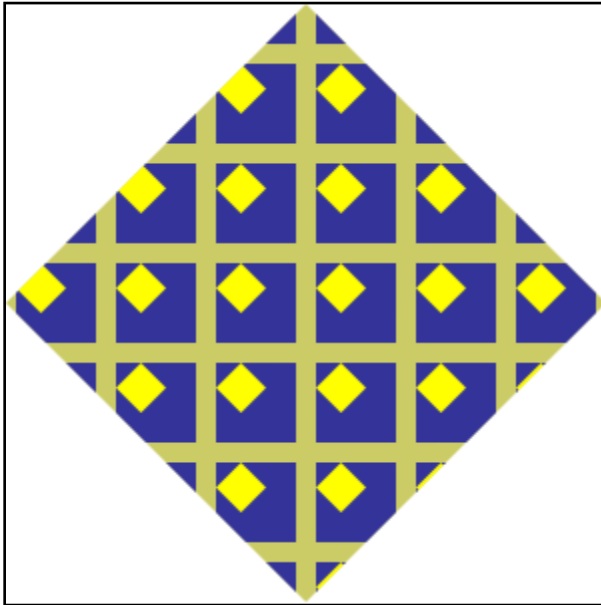
20 The following examples demonstrate how adjusting the viewbox and viewport can affect output.

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*

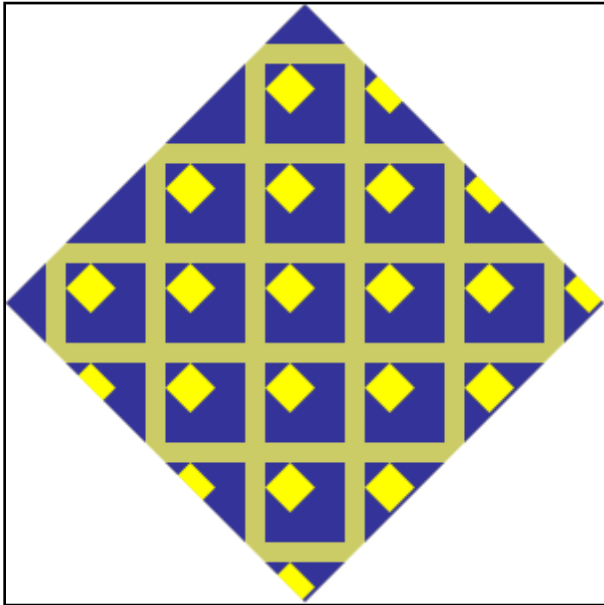
5 By adjusting the viewport, the position of the tiles within the image can be changed:

```

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*

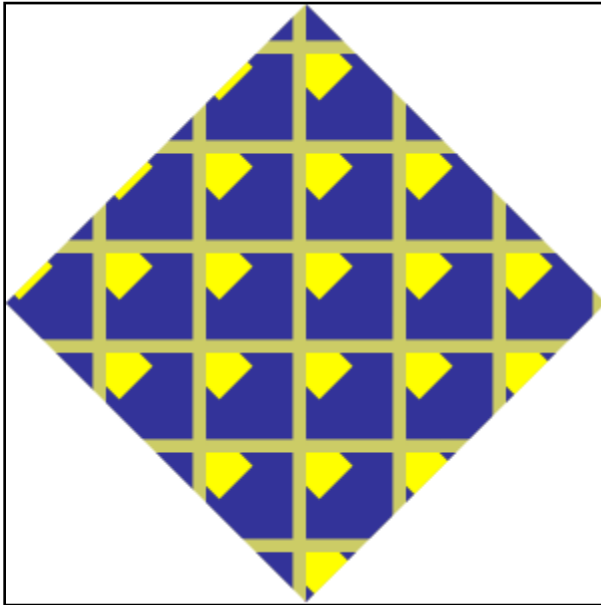
5 The following markup uses a smaller window on the viewbox to zoom in on each tile:

```

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*

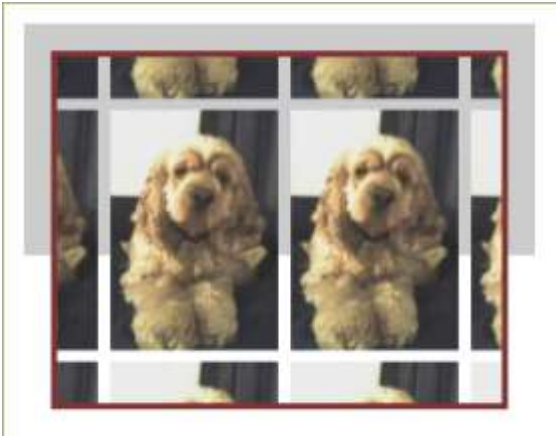
5 An image brush can specify a tile with the Viewbox attribute that exceeds the size of the image  
6 it uses, including negative values, as shown below.

```

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.

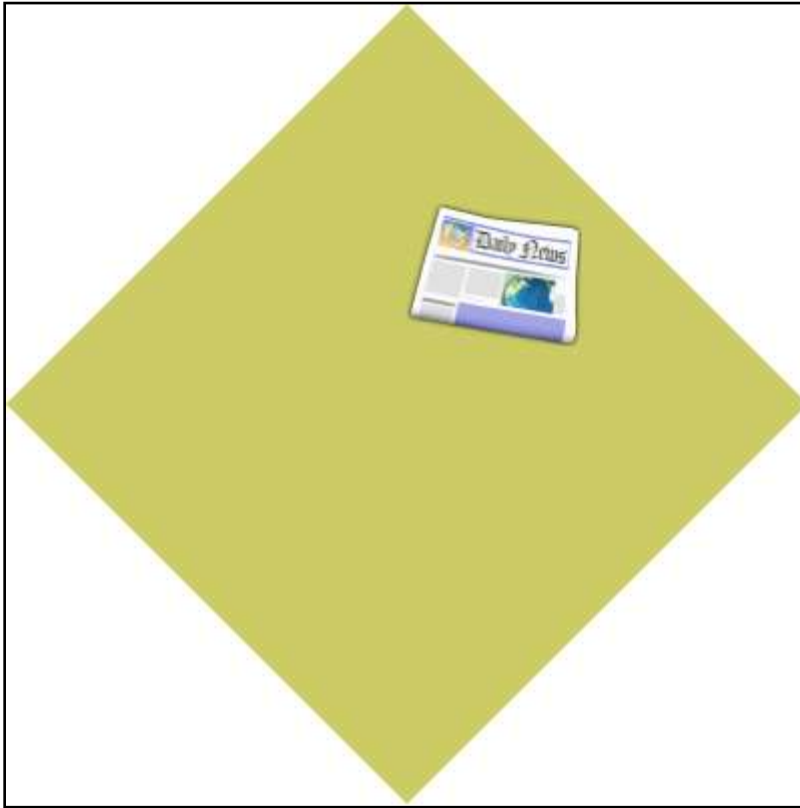
- 9 *Example 13-9. Image brush with TileMode value of None*

```

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]*

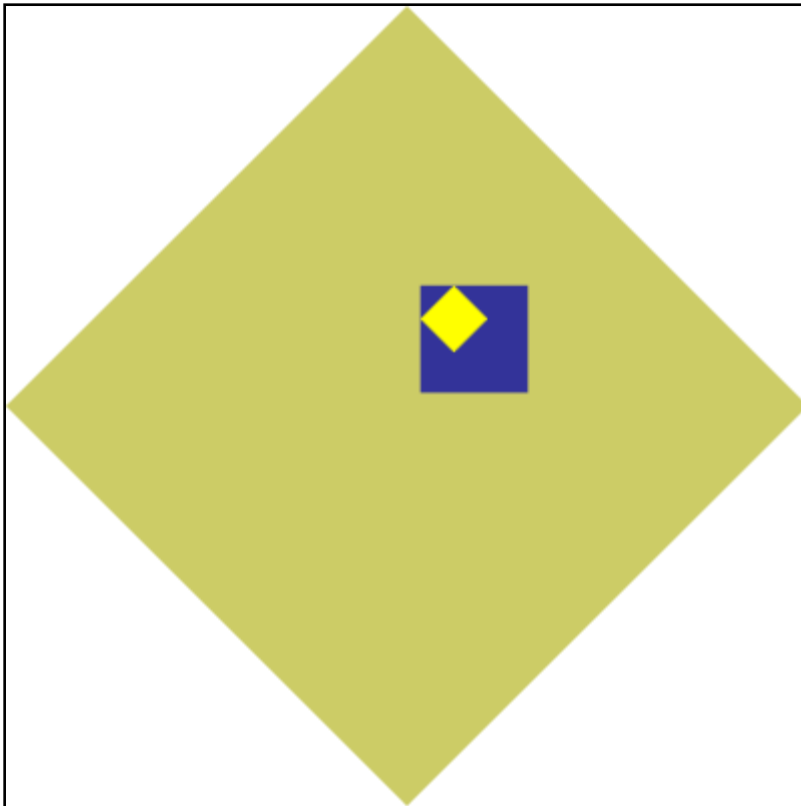
1 *Example 13-10. Visual brush with TileMode value of None*

```

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]*

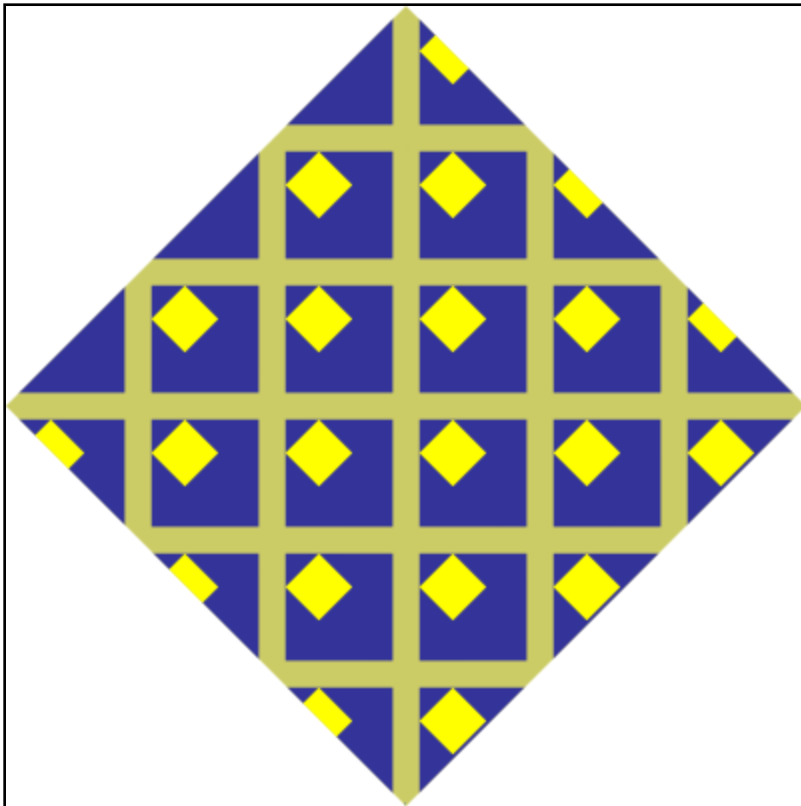
1 *Example 13-12. Visual brush with a TileMode value of Tile*

```

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]*

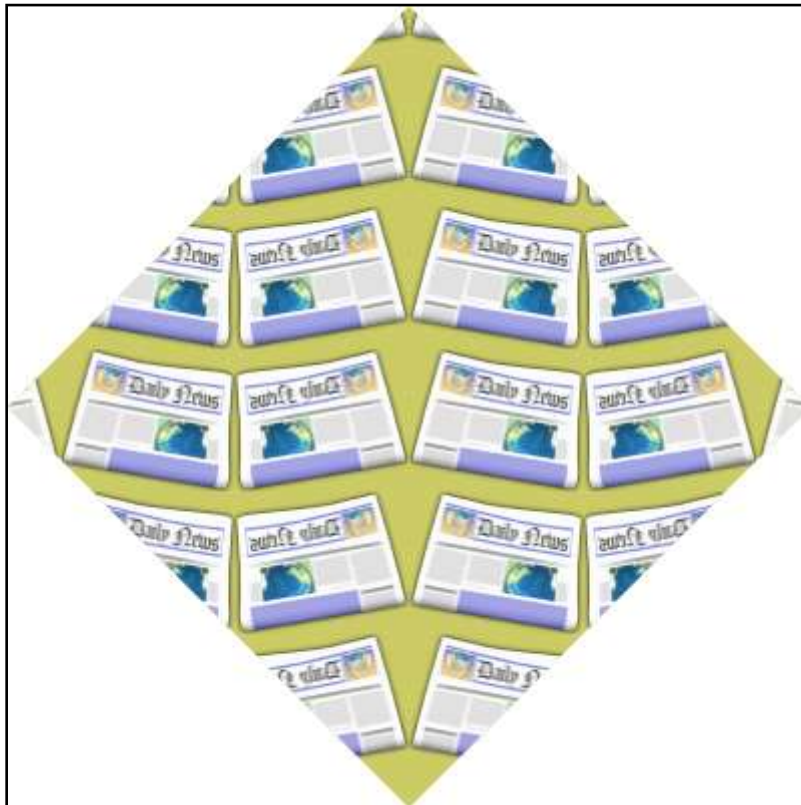
### 1 13.4.2.3 FlipX

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]*

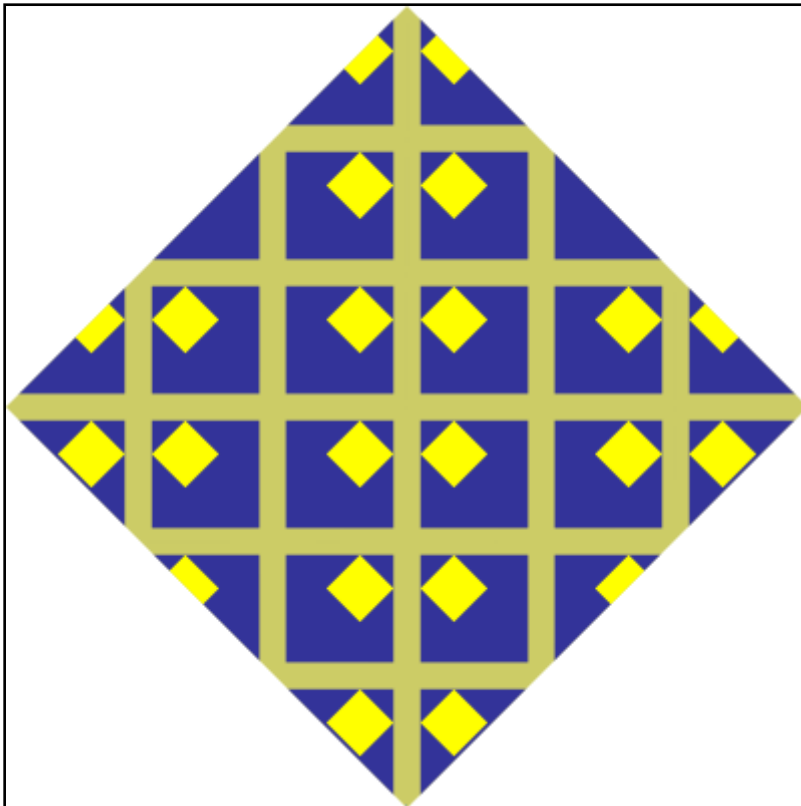
1 *Example 13-14. Visual brush with a TileMode value of FlipX*

```

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]*

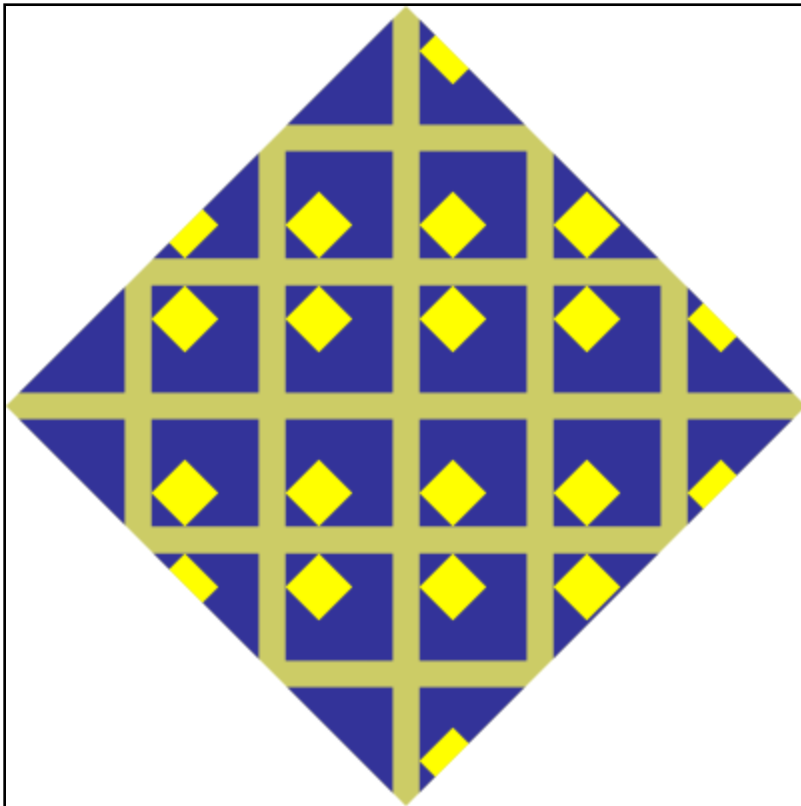
1 *Example 13-16. Visual Brush with a TileMode value of FlipY*

```

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]*

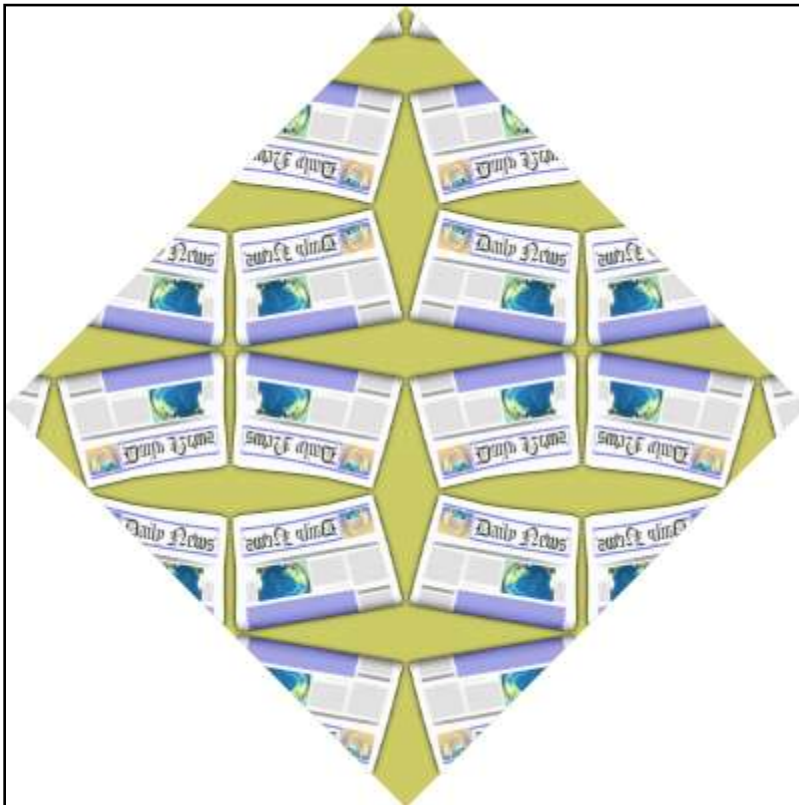
### 1 13.4.2.5 FlipXY

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]*

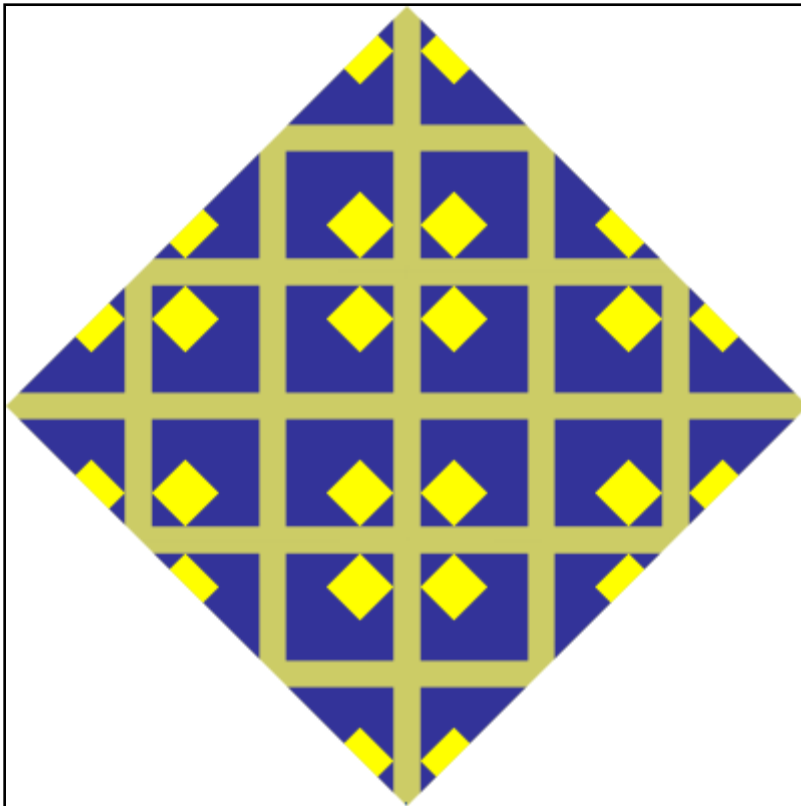
1 *Example 13-18. Visual brush with a TileMode value of FlipXY*

```

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:



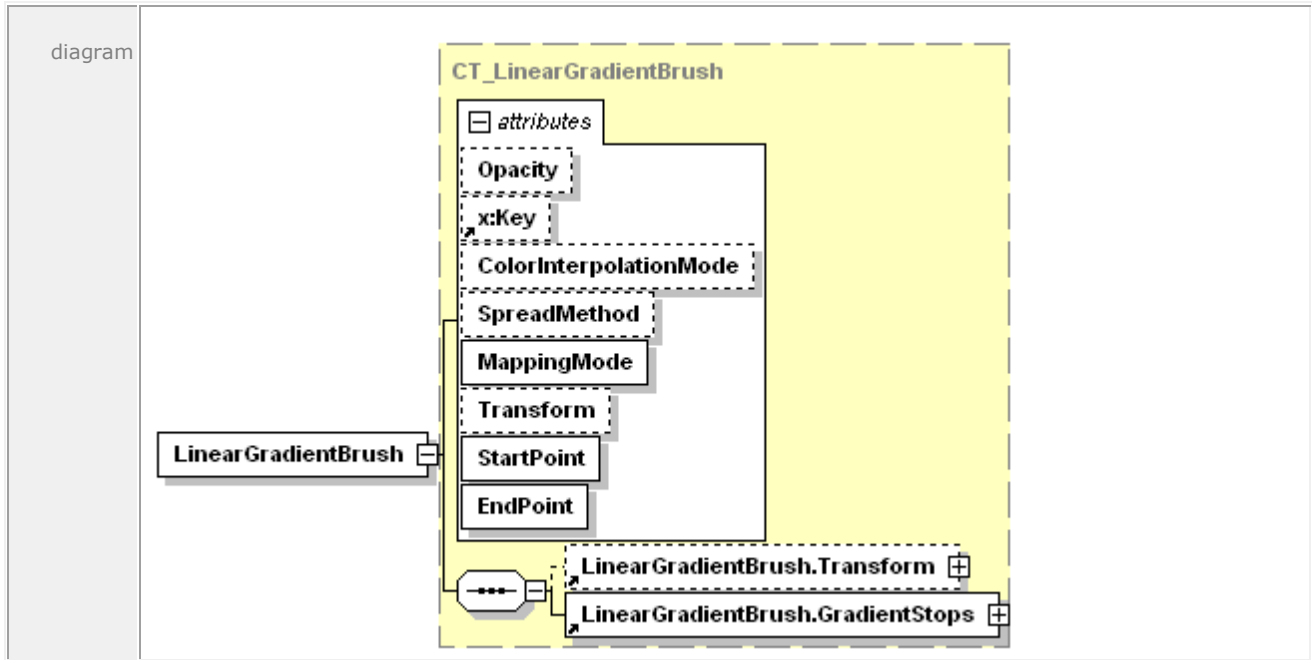
24

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	<a href="#">ST_SpreadMethod</a>		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	<a href="#">ST_MappingMode</a>	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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_Point</a>	required			Specifies the starting point of the linear gradient.
	EndPoint	<a href="#">ST_Point</a>	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  
5 gradient:

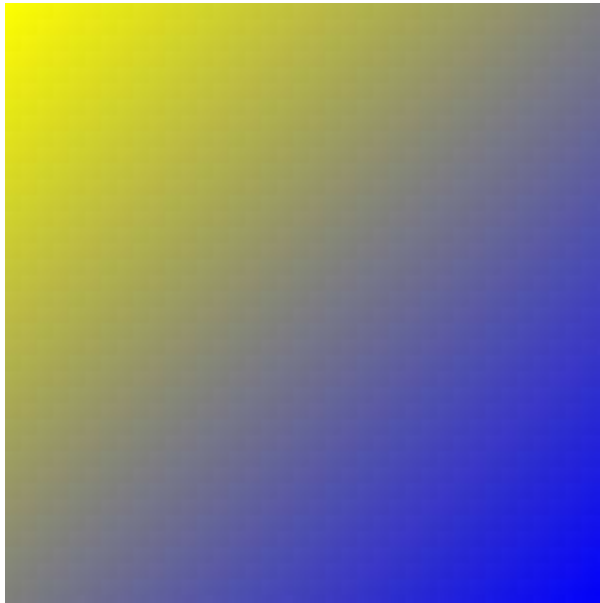
```
6     <Path>
7         <Path.Fill>
```

```

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. For details see §18.3.2.

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

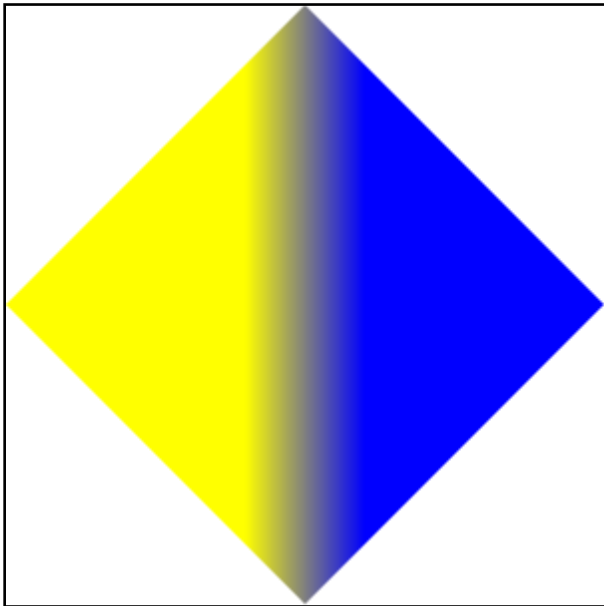
```

```

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*

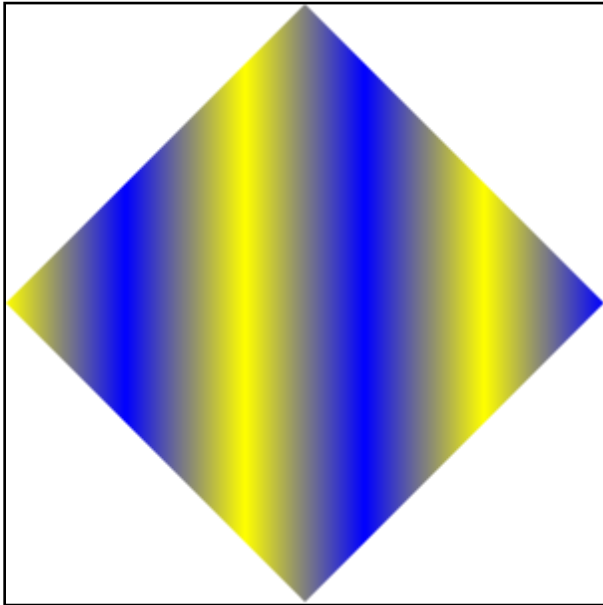
16 In this method, the gradient stops are replayed in reverse order repeatedly to cover the fill  
17 area.

```

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]*

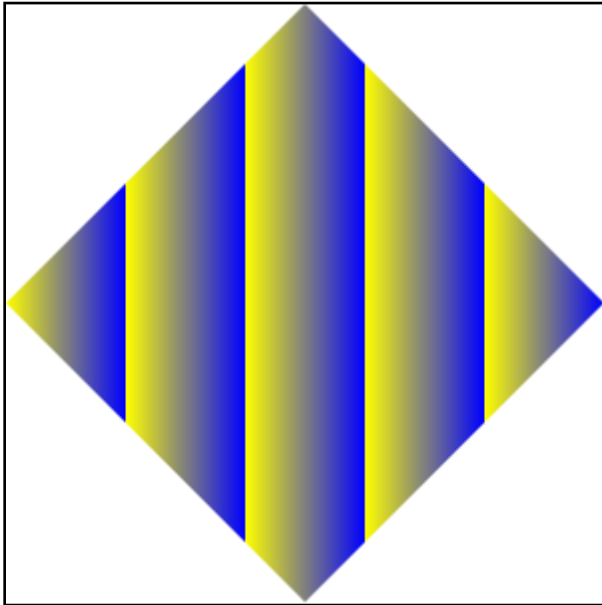
4 *Example 13–22. Linear gradient brush with a SpreadMethod value of Repeat*

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>
  
```

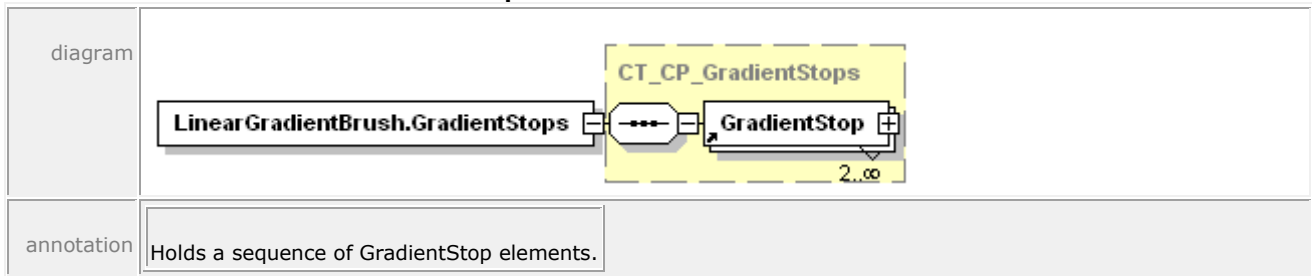
1 This markup is rendered as follows:



2  
3 *end example]*

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

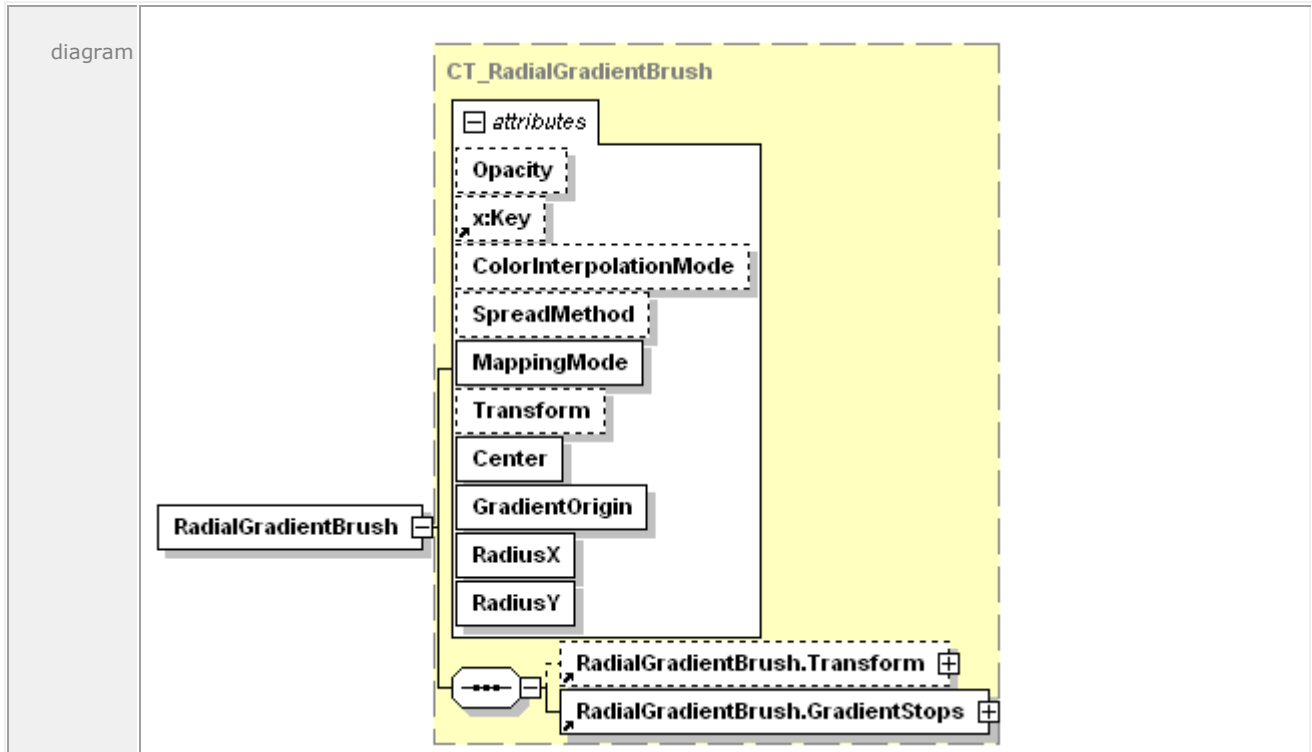
5 element **LinearGradientBrush.GradientStops**



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. The gamma adjustment should not be applied to the alpha component, if specified. Valid values are SRgbLinearInterpolation and	

					ScRgbLinearInterpolation.
SpreadMethod	<a href="#">ST_SpreadMethod</a>		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	<a href="#">ST_MappingMode</a>	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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_Point</a>	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	<a href="#">ST_Point</a>	required			Specifies the origin point of the radial gradient.
RadiusX	<a href="#">ST_GEZero</a>	required			Specifies the radius in the x dimension of the ellipse which defines the radial gradient.
RadiusY	<a href="#">ST_GEZero</a>	required			Specifies the radius in the y dimension of the ellipse which defines the radial gradient.
annotation	Fills a region with a radial gradient.				

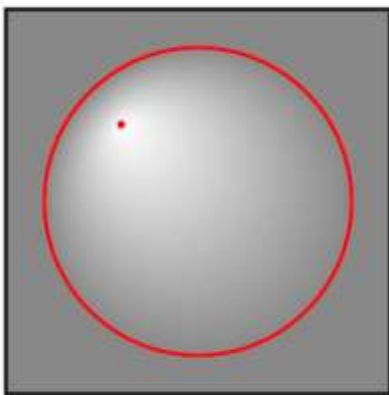


1 Radial gradient brushes are similar to linear gradient brushes. However, whereas a linear  
 2 gradient brush has a start point and end point to define the gradient vector, a radial gradient  
 3 brush has an ellipse (defined by the center,  $x$  radius, and  $y$  radius) and a gradient origin. The  
 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. A gradient stop with an offset at 0.0  
 6 defines the color at the gradient origin.

7 For details about computing a radial gradient, see §18.3.3.

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

9 The following figure is a radial gradient that transitions from white to gray. The outside ellipse  
 10 represents the gradient ellipse while the dot denotes the gradient origin. This gradient has a  
 11 SpreadMethod value of Pad.



12

13 *end example]*

14 *Example 13–24. RadialGradientBrush usage*

15 The following markup describes a page with a rectangular path that is filled with a radial  
 16 gradient:

```

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

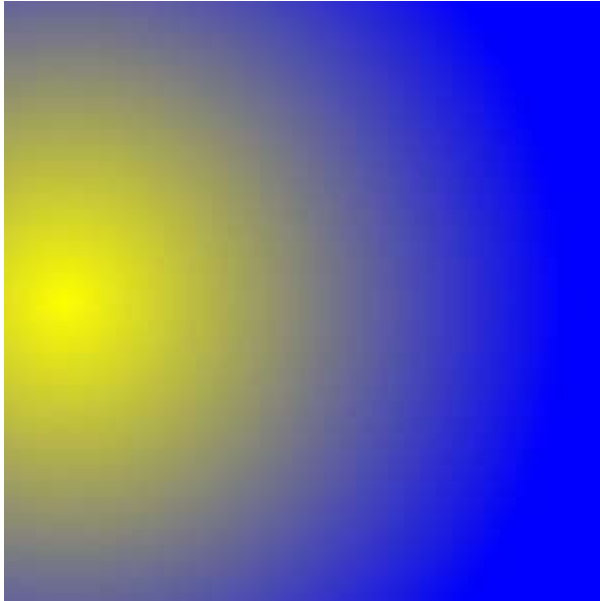
```

```

1         </PathFigure>
2     </PathGeometry>
3 </Path.Data>
4 </Path>

```

5 This markup is rendered as follows:



6  
7 *end example]*

### 8 **13.6.1 SpreadMethod Attribute**

9 The SpreadMethod attribute describes the fill of areas beyond the ellipse described by the center,  
10 x radius, and y radius of the radial gradient brush. Valid values are Pad, Reflect, and Repeat.  
11 For details see §18.3.3.

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

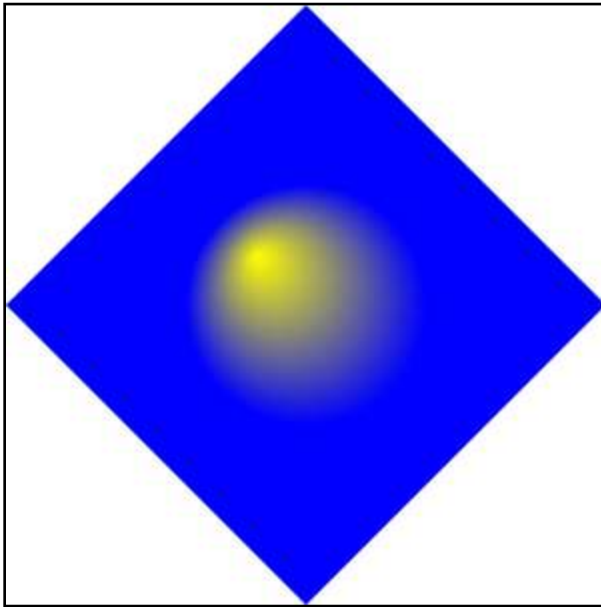
13 In the following markup, the last color is used to cover the fill area outside the ellipse.

```

14 <Path Data="M 150,0 L 300,150 L 150,300 L 0,150 Z">
15 <Path.Fill>
16 <RadialGradientBrush
17     MappingMode="Absolute"
18     Center="150,150"
19     GradientOrigin="125,125"
20     RadiusX="60"
21     RadiusY="60"
22     SpreadMethod="Pad">
23 <RadialGradientBrush.GradientStops>
24     <GradientStop Color="#FFFF00" Offset="0.0" />
25     <GradientStop Color="#0000FF" Offset="1.0" />
26 </RadialGradientBrush.GradientStops>
27 </RadialGradientBrush>
28 </Path.Fill>
29 </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*

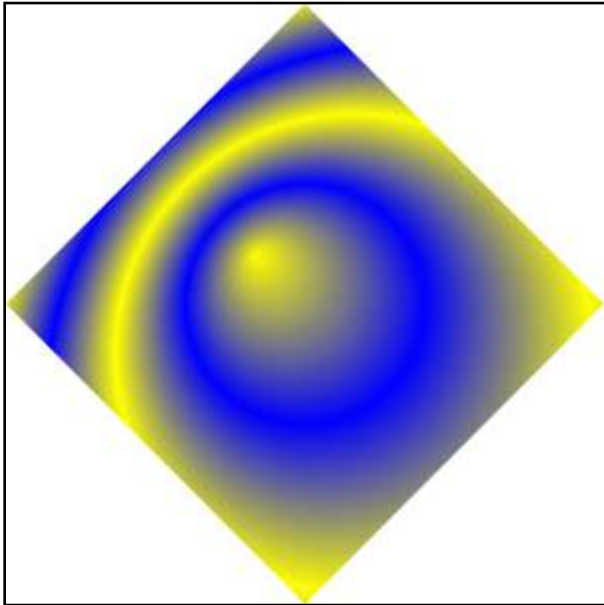
5 In the following markup, the gradient stops are replayed in reverse order repeatedly to cover  
6 the fill area.

```

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*

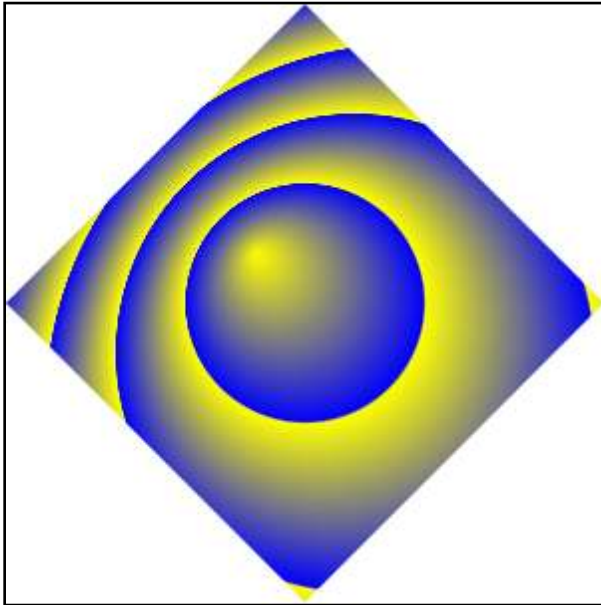
5 In the following markup, the gradient stops are repeated in order until the fill area is covered.

```

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**

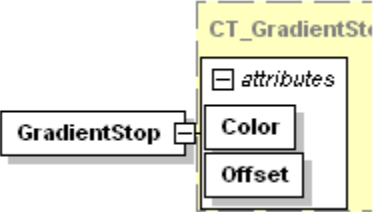
5 element **RadialGradientBrush.GradientStops**

<p>diagram</p>	
<p>annotation</p>	<p>Holds a sequence of &lt;GradientStop&gt; elements.</p>

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	<a href="#">ST_Color</a>	required			Specifies the gradient stop color.
	Offset	<a href="#">ST_Double</a>	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 as the child element (see §14.5).

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]*

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

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

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



1 This markup is rendered as follows:



2

3 *end example]*



## 1 14. Common Properties

2 Several OpenXPS 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 §18.5.

6 *Table 14–1. Common property attributes*

<b>Name</b>	<b>Applies to</b>	<b>Description</b>
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 [M7.12].

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].

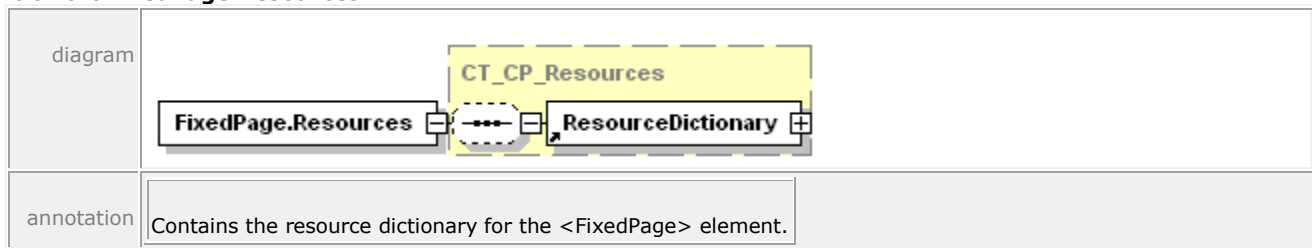
5 The <Canvas.Resources> or <FixedPage.Resources> property elements MUST precede any  
 6 property elements of the <Canvas> or <FixedPage> elements [M2.72]. Likewise, they MUST  
 7 precede any path, glyphs, or canvas children of the <Canvas> or <FixedPage> elements  
 8 [M7.14].

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**



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

```

20 <FixedPage Width="816" Height="1056" xml:lang="en-US"
21   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
22   xmlns:x=
23
24 "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
25 ctionary-key">
26   <FixedPage.Resources>
27     <ResourceDictionary>
28       <PathGeometry x:Key="Rectangle">
29         <PathFigure StartPoint="20,20" IsClosed="true">
30           <PolyLineSegment Points="120,20 120,70 20,70" />
31         </PathFigure>
32       </PathGeometry>
33     </ResourceDictionary>
34   </FixedPage.Resources>
35   <Path Stroke="#000000"
36     StrokeThickness="1"
37     Data="{StaticResource Rectangle}" />
38 </FixedPage>

```

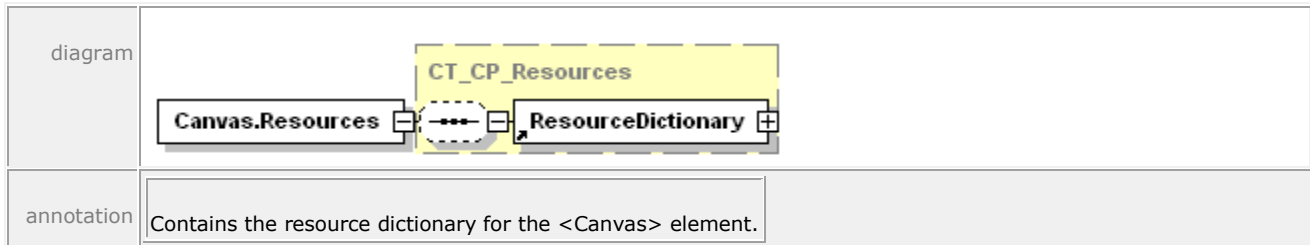
1 This markup is rendered as follows:



2  
3 *end example]*

#### 4 **14.2.2 <Canvas.Resources> Element**

5 element **Canvas.Resources**



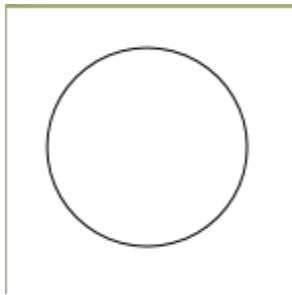
6  
7 *Example 14-2. <Canvas.Resources> usage*

```

8   <Canvas
9     xmlns:x=
10
11 |  "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
12 |  ctionary-key">
13   <Canvas.Resources>
14     <ResourceDictionary>
15       <PathGeometry x:Key="Circle">
16         <PathFigure StartPoint="20,70">
17           <ArcSegment
18             Point="120,70"
19             Size="50,50"
20             RotationAngle="0"
21             IsLargeArc="true"
22             SweepDirection="Clockwise" />
23           <ArcSegment
24             Point="20,70"
25             Size="50,50"
26             RotationAngle="0"
27             IsLargeArc="true"
28             SweepDirection="Clockwise" />
29         </PathFigure>
30       </PathGeometry>
31     </ResourceDictionary>
32   </Canvas.Resources>
33   <Path Stroke="#000000"
34     StrokeThickness="1"
35     Data="{StaticResource Circle}" />
36 </Canvas>

```

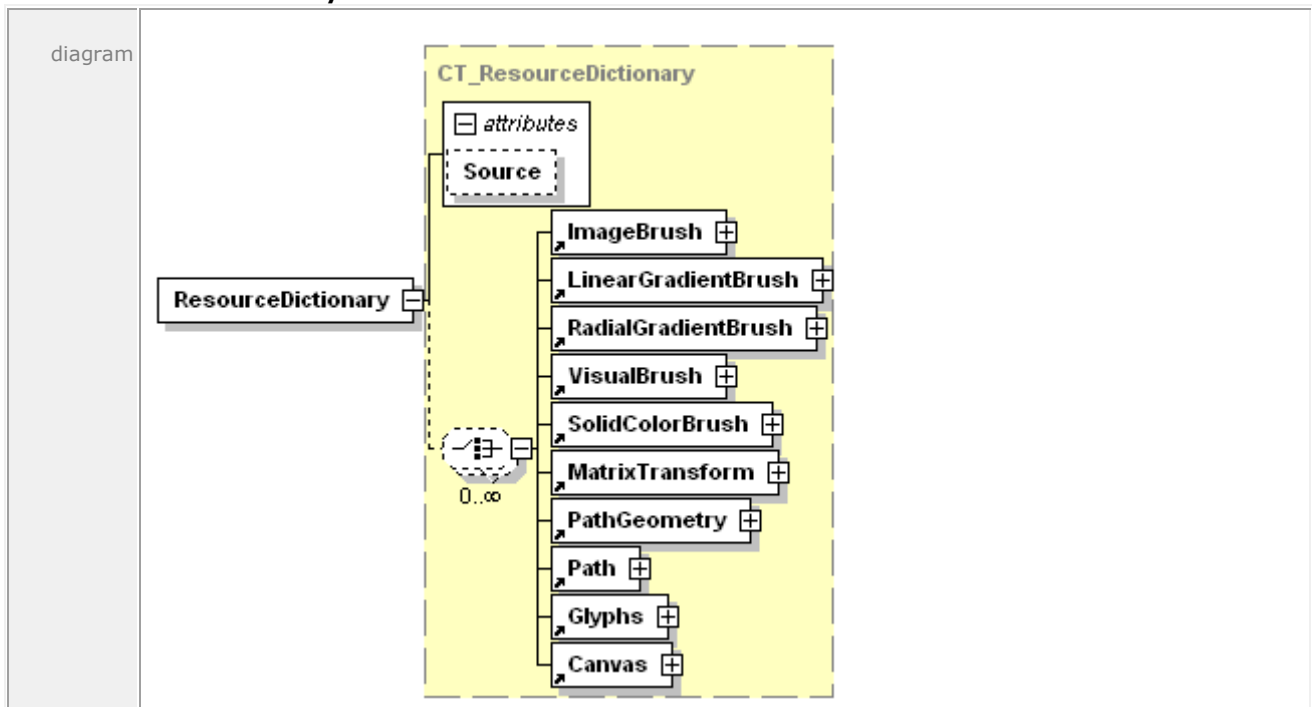
1 This markup is rendered as follows:



2  
3 *end example]*

4 **14.2.3 <ResourceDictionary> Element**

5 element **ResourceDictionary**



attributes	Name	Type	Use	Default	Fixed	Annotation
	Source	xs:anyURI				Specifies the URI of a part containing markup for a resource dictionary. The URI MUST refer to a part in the package [M2.1].
annotation	Defines a set of reusable resource definitions that can be used as property values in the fixed page markup.					

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

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

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

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

15 *Example 14–3. Resource dictionary markup*

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

```

17 <Canvas
18     xmlns:x=
19
20 | "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
21 ctionary-key">
22     <Canvas.Resources>
23         <ResourceDictionary>
24             <PathGeometry x:Key="Rectangle">
25                 <PathFigure StartPoint="20,20" IsClosed="true">
26                     <PolyLineSegment Points="120,20 120,70 20,70" />
27                 </PathFigure>
28             </PathGeometry>
29             <PathGeometry x:Key="Circle">
30                 <PathFigure StartPoint="20,70">
31                     <ArcSegment
32                         Point="120,70"
33                         Size="50,50"
34                         RotationAngle="0"
35                         IsLargeArc="true"
36                         SweepDirection="Clockwise" />
37                     <ArcSegment
38                         Point="20,70"
39                         Size="50,50"
40                         RotationAngle="0"
41                         IsLargeArc="true"
42                         SweepDirection="Clockwise" />
43                 </PathFigure>
44             </PathGeometry>
45         </ResourceDictionary>
46     </Canvas.Resources>
47     <Path Data="{StaticResource Rectangle}">
48         <Path.Fill>
49             <SolidColorBrush Color="#FF0000" />
50         </Path.Fill>
51 </Path>

```



1       </Canvas>

2   *end example]*

### 3   **14.2.3.1 Remote Resource Dictionaries**

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

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

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

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

21   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 [M7.7].

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

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

```

27       <!-- Contents of /resource.xaml -->
28       <ResourceDictionary
29       xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
30       xmlns:x=
31
32       "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
33       ctionary-key">
34        <PathGeometry x:Key="Rectangle">
35          <PathFigure StartPoint="20,20" IsClosed="true">
36            <PolyLineSegment Points="120,20 120,70 20,70" />
37          </PathFigure>
38        </PathGeometry>
39        <PathGeometry x:Key="Circle">
40          <PathFigure StartPoint="20,70">
41            <ArcSegment
42              Point="120,70"
43              Size="50,50"
44              RotationAngle="0"
45              IsLargeArc="true"
46              SweepDirection="Clockwise" />

```

```

1         <ArcSegment
2             Point="20,70"
3             Size="50,50"
4             RotationAngle="0"
5             IsLargeArc="true"
6             SweepDirection="Clockwise" />
7     </PathFigure>
8 </PathGeometry>
9 </ResourceDictionary>

```

10 The following markup references the previously defined resource dictionary:

```

11 <Canvas
12     xmlns:x=
13
14     "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
15     ctionary-key">
16     <Canvas.Resources>
17         <ResourceDictionary Source="/resource.xaml"/>
18     </Canvas.Resources>
19     <Path Data="{StaticResource Rectangle}">
20         <Path.Fill>
21             <SolidColorBrush Color="#FF0000" />
22         </Path.Fill>
23     </Path>
24 </Canvas>

```

25 *end example]*

#### 26 14.2.4 Resource References

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

```
28     {StaticResource key}
```

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

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

35 [A consumer SHOULD instantiate](#) ~~It is considered~~ an error condition if a static resource reference  
36 cannot be resolved, or if it *can* be resolved but the resource type does not match the usage at  
37 the location of reference [\[S7.1\]](#).

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

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

```

41 <Canvas
42     xmlns:x=
43
44     "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
45     ctionary-key">

```

```

1      <Canvas.Resources>
2          <ResourceDictionary>
3              <PathGeometry x:Key="Rectangle">
4                  <PathFigure StartPoint="20,20" IsClosed="true">
5                      <PolyLineSegment Points="120,20 120,70 20,70" />
6                  </PathFigure>
7              </PathGeometry>
8          </ResourceDictionary>
9      </Canvas.Resources>
10     <Path Data="{StaticResource Rectangle}">
11         <Path.Fill>
12             <SolidColorBrush Color="#FF0000" />
13         </Path.Fill>
14     </Path>
15 </Canvas>

```

16 *end example]*

### 17 **14.2.5 Scoping Rules for Resolving Resource References**

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

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

25 A resource definition MAY reference another resource defined prior to the point of reference,  
 26 including a resource previously defined within the same resource dictionary [O7.2]. If a  
 27 resource definition references another resource, the reference MUST be resolved in the context  
 28 of the resource definition, not in the context of the resource use [M7.9].

29 To find a resource, the nearest parent or ancestor canvas or fixed page is searched. If the  
 30 desired name is not defined in the initially searched resource dictionary, then the next-nearest  
 31 parent or ancestor canvas or fixed page is searched. [A consumer SHOULD instantiate an](#)  
 32 [error condition](#) occurs if the search has continued to the root <FixedPage> element and a  
 33 specified resource has not been found [\[S7.2\]](#). This search occurs only within the containing  
 34 FixedPage part.

35 *Example 14–6. Using scoping rules*

```

36     <FixedPage
37         xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
38         xmlns:x=
39
40         "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedi
41         ctionary-key"
42         Height="1056" Width="816" xml:lang="en-US">
43         <FixedPage.Resources>
44             <ResourceDictionary>
45                 <SolidColorBrush x:Key="FavoriteColorFill" Color="#808080" />
46             </ResourceDictionary>
47         </FixedPage.Resources>
48     </Canvas>

```

```

1      <Canvas.Resources>
2          <ResourceDictionary>
3              <SolidColorBrush x:Key="FavoriteColorFill"
4                  Color="#000000" />
5          </ResourceDictionary>
6      </Canvas.Resources>
7      <!-- The following path is filed with color #000000 -->
8      <Path Fill="{StaticResource FavoriteColorFill}">
9          <Path.Data>
10             ...
11         </Path.Data>
12     </Path>
13 </Canvas>
14 <!-- The following path is filed with color #000000 -->
15 <Path Fill="{StaticResource FavoriteColorFill}">
16     <Path.Data>
17         ...
18     </Path.Data>
19 </Path>
20 </Canvas>
21 </Canvas>
22 <!-- The following path is filled with color #808080 -->
23 <Path Fill="{StaticResource FavoriteColorFill}">
24     <Path.Data>
25         ...
26     </Path.Data>
27 </Path>
28 </FixedPage>

```

29 *end example]*

### 30 **14.2.6 Support for Markup Compatibility**

31 If a resource dictionary contains Markup Compatibility and Extensibility elements and  
32 attributes, the processing of the Markup Compatibility and Extensibility markup **MUST** occur in  
33 the context of the definition of the resource dictionary, not in the context of resource references  
34 [M2.10].

---

## 35 **14.3 Clipping**

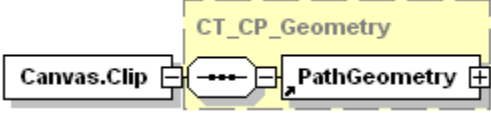
36 The Clip property specifies a geometric area that restricts the rendered region of an element.

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

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

### 40 **14.3.1 <Canvas.Clip> Element**

41 element **Canvas.Clip**

diagram	
annotation	Limits the rendered region of the element.

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

3 *Example 14–7. Canvas clip markup and rendering*

```

4     <Canvas>
5         <Canvas.Clip>
6             <PathGeometry>
7                 <PathFigure StartPoint="25,25" IsClosed="true">
8                     <PolyLineSegment Points="60,25 70,60 80,25 115,25
9                         115,115 80,115 70,80 60,115 25,115" />
10                </PathFigure>
11            </PathGeometry>
12        </Canvas.Clip>
13        <Path Fill="#9999CC">
14            <Path.Data>
15                <PathGeometry>
16                    <PathFigure StartPoint="20,70">
17                        <ArcSegment
18                            Point="120,70"
19                            Size="50,50"
20                            RotationAngle="0"
21                            IsLargeArc="true"
22                            SweepDirection="Clockwise" />
23                    <ArcSegment
24                        Point="20,70"
25                        Size="50,50"
26                        RotationAngle="0"
27                        IsLargeArc="true"
28                        SweepDirection="Clockwise" />
29                </PathFigure>
30            </PathGeometry>
31        </Path.Data>
32    </Path>
33 </Canvas>

```

34 This markup is rendered as follows:

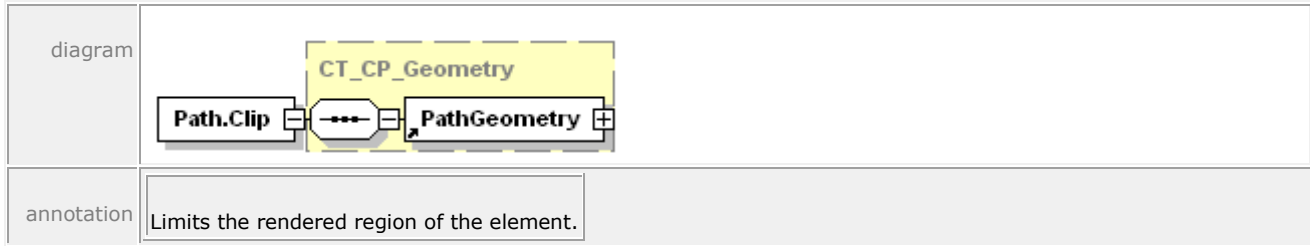


35

1 *end example]*

### 2 **14.3.2 <Path.Clip> Element**

3 element **Path.Clip**



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

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

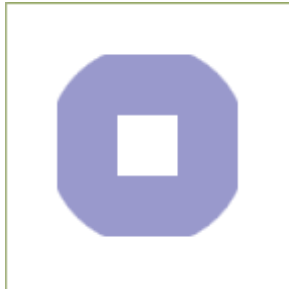
6 The following markup describes a complex clipping behavior:

```

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

```

1 This markup is rendered as follows:



2

3 *end example]*

#### 4 **14.3.3 <Glyphs.Clip> Element**

5 element **Glyphs.Clip**

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

6 *Example 14–9. <Glyphs.Clip> usage*

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

```

8   <Glyphs
9     Fill="#000000"
10    Clip="M 0,0 L 180,0 L 180,140 L 0,140 Z M 20,60 L 140,60 L 140,80
11         L 20,80 Z"
12    OriginX="20"
13    OriginY="130"
14    UnicodeString="N"
15    FontRenderingEmSize="170"
16    FontUri="../Resources/Fonts/Timesbd.ttf" />

```

17 This markup is rendered as follows:



18

19 *end example]*

**14.4 Positioning Content**

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

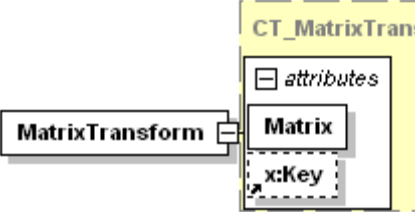
Elements are positioned relative to the current origin (0,0) of the coordinate space. The current origin can be moved by setting the RenderTransform property of a canvas, path, or glyph. The render transformation establishes a new coordinate frame for all children of the parent element.

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

The RenderTransform and Transform properties both specify an affine matrix transformation to the local coordinate space, using the <MatrixTransform> element as their value. An abbreviated matrix transformation syntax MAY be used to specify a RenderTransform or Transform attribute value [O7.7].

**14.4.1 <MatrixTransform> Element**

element **MatrixTransform**

diagram						
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.					

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



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

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

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

6 M11,M12,M21,M22,OffsetX,OffsetY

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

10 *Example 14-10. Matrix scaling*

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

11 *end example*]

12 *Example 14-11. Matrix reversing the x axis*

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

13 *end example*]

14 *Example 14-12. Matrix reversing the y axis*

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

15 *end example*]

16 *Example 14-13. Matrix skewing*

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

17 *end example*]

1 *Example 14-14. Matrix Rotating*

$$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2 *end example]*

3 *Example 14-15. Matrix positioning*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \text{OffsetX} & \text{OffsetY} & 1 \end{bmatrix}$$

4 *end example]*

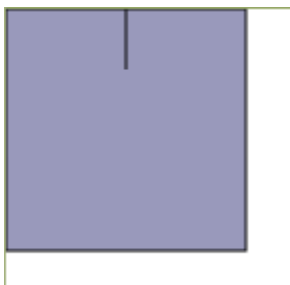
5 *Example 14-16. <MatrixTransform> usage*

6 The following markup describes a box (with the top edge marked) that is rotated 90° and  
7 shifted 50 units down and to the right:

```
8     <Path
9         Stroke="#000000"
10        Fill="#9999BB"
11        Data="M 0,0 L 60,0 L 60,25 L 60,0 L 120,0 L 120,120 L 0,120 Z">
12        <Path.RenderTransform>
13            <MatrixTransform Matrix="0,1,-1,0,170,50" />
14        </Path.RenderTransform>
15    </Path>
```

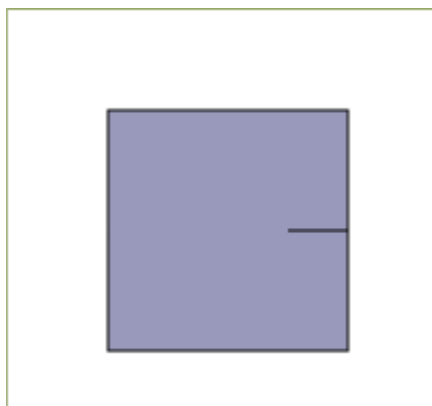
16 Since the x origin has been shifted, the overall box must be additionally shifted the width of the  
17 box to achieve the desired visual effect.

18 Before the render transformation, the box appears like this:



19

1 After the render transformation, the box appears like this:



2

3 *end example]*

1 *Example 14-17. Using abbreviated matrix transformation syntax*

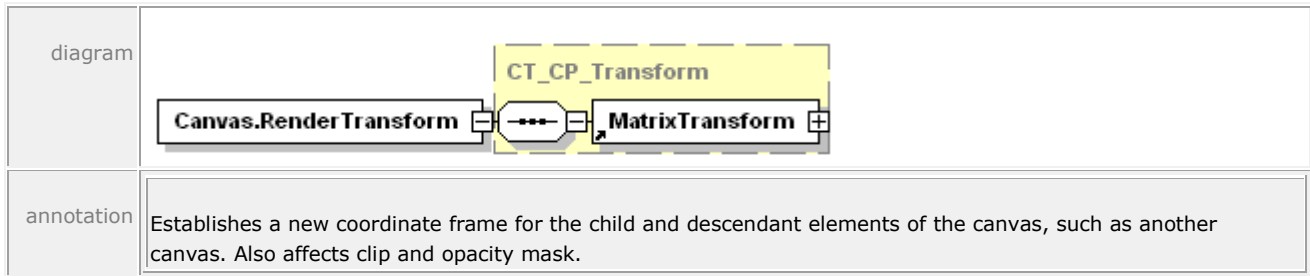
2 The following markup uses abbreviated syntax to produce the above image:

```
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:

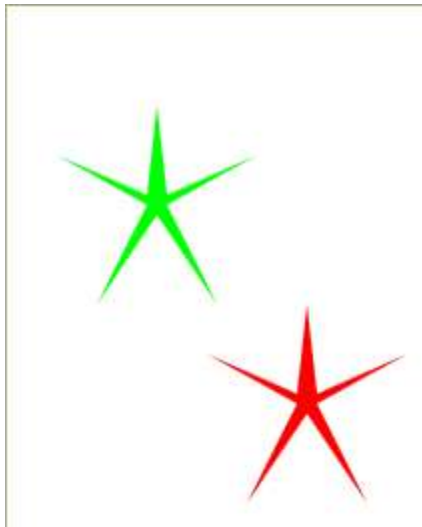
```
14   <Canvas>
15     <Canvas.Resources>
16       <ResourceDictionary>
17         <PathGeometry x:Key="StarFish">
18           <PathFigure StartPoint="50,0" IsClosed="true">
19             <PolyLineSegment Points="55,45 100,25 55,50 80,100 50,55
20               20,100 45,50 0,25 45,45" />
21           </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**

diagram	
annotation	<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 &lt;Path.Data&gt; property element.</p>

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" >

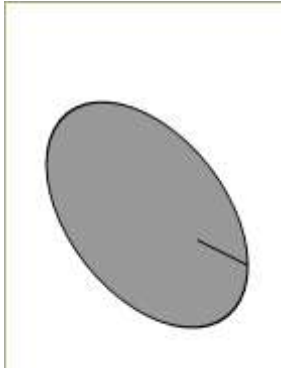
```

```

1      <Path.RenderTransform>
2      <MatrixTransform Matrix="1,0.5,0,1,0,0" />
3      </Path.RenderTransform>
4  </Path>

```

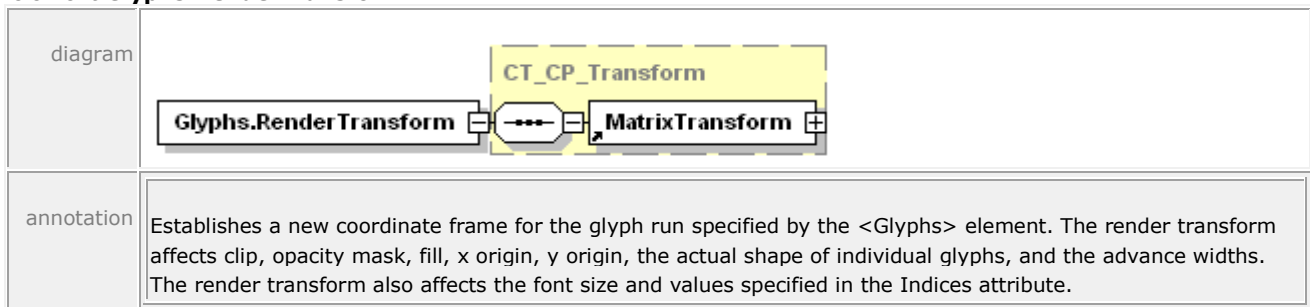
5 This markup is rendered as follows:



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*

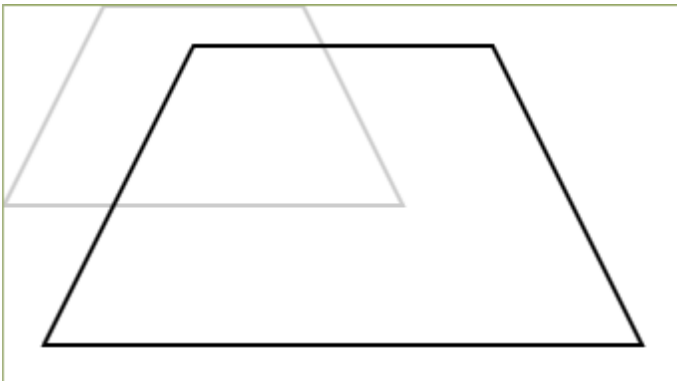
7 The following markup demonstrates a simple 150% zoom and positional transformation:

```

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]*

#### 6 **14.4.6 <ImageBrush.Transform> Element**

- 7 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>

- 8 The Transform property can result in a non-rectangular (that is, skewed) viewport that defines  
 9 the tile shape. In this circumstance, tile mode operations (FlipX, FlipY, and FlipXY) are treated  
 10 as if the tile was rectangular, a larger tile was constructed from a 2-by-2 arrangement of  
 11 regular tiles, the skew transform was applied afterward, and the new non-rectangular tile was  
 12 tiled with adjacent edges and without flipping.

- 13 *Example 14-22. <ImageBrush.Transform> usage*

- 14 The following markup describes an image rotated 20° and repositioned within a path. The path  
 15 itself remains untransformed; the viewport of the image brush is transformed instead.

```

16 <Path
17   StrokeThickness="5"
18   Stroke="#996666"
19   StrokeLineJoin="Round"
20   Data="M 25,25 L 350,25 L 355,250 L 25,250 Z">
21   <Path.Fill>
22     <ImageBrush
23       ImageSource="dog.jpg"
24       TileMode="Tile"
25       Viewbox="0,0,270,423"
26       ViewboxUnits="Absolute"
27       Viewport="75,75,90,125"

```

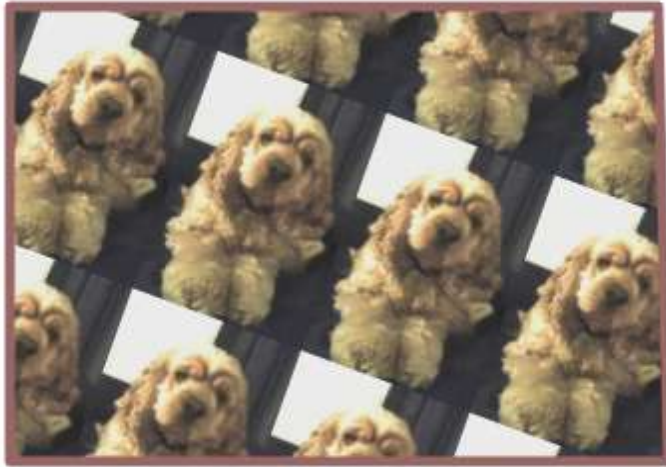


```

1      ViewportUnits="Absolute" >
2      <ImageBrush.Transform>
3          <MatrixTransform Matrix=".939,.342,-.342,.939,0,-80" />
4      </ImageBrush.Transform>
5  </ImageBrush>
6  </Path.Fill>
7  </Path>

```

8 This markup is rendered as follows:



9

10 *end example]*

### 11 14.4.7 <VisualBrush.Transform> Element

12 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>

13 The Transform property can result in a non-rectangular (that is, skewed) viewport that defines  
14 the tile shape. In this circumstance, tile mode operations (FlipX, FlipY, and FlipXY) are treated  
15 as if the tile was rectangular, a larger tile was constructed from a 2-by-2 arrangement of  
16 regular tiles, the skew transform was applied afterward, and the new non-rectangular tile was  
17 tiled with adjacent edges and without flipping.

18 *Example 14–23. <VisualBrush.Transform> usage*

19 The following markup describes a solid background and vertical pinstripe rotated 45° to fill a  
20 frame:

```

1 <Path
2   StrokeThickness="5"
3   Stroke="#336666"
4   StrokeLineJoin="Round"
5   Data="M 25,25 L 365,25 L 365,250 L 25,250 Z M 70,70 L 320,70
6     L 320,205 L 70,205 Z">
7   <Path.Fill>
8     <VisualBrush
9       TileMode="Tile"
10      Viewbox="0,0,60,100"
11      ViewboxUnits="Absolute"
12      Viewport="25,25,50,50"
13      ViewportUnits="Absolute">
14     <VisualBrush.Transform>
15       <MatrixTransform Matrix=".707,.707,-.707,.707,0,0" />
16     </VisualBrush.Transform>
17     <VisualBrush.Visual>
18       <Canvas>
19         <Path
20           Fill="#99CCCC"
21           Data="M 0,0 L 60,0 L 60,100 L 0,100 Z" />
22         <Path
23           Stroke="#336666"
24           Data="M 0,0 L 0,100 M 20,0 L 20,100 M 40,0 L 40,100
25             M 60,0 L 60,100 M 80,0 L 80,100" />
26         </Canvas>
27       </VisualBrush.Visual>
28     </VisualBrush>
29   </Path.Fill>
30 </Path>

```

31 This markup is rendered as follows:



32

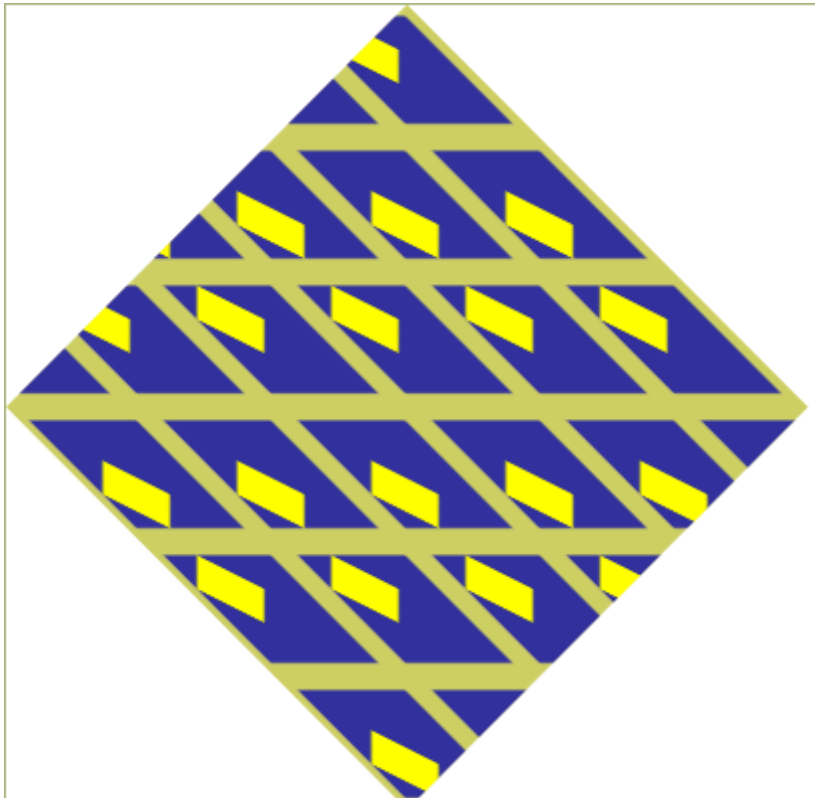
33 *end example]*

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>

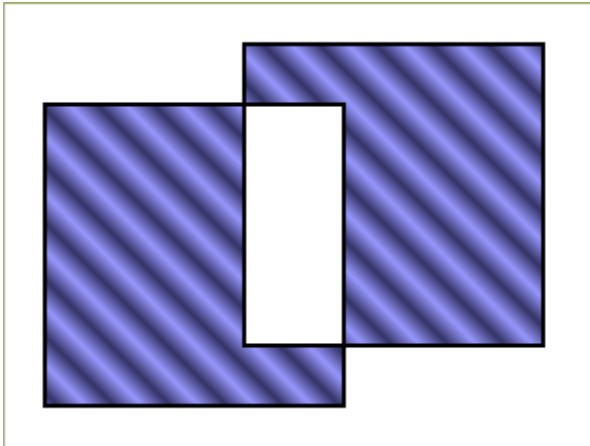
```

```

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>

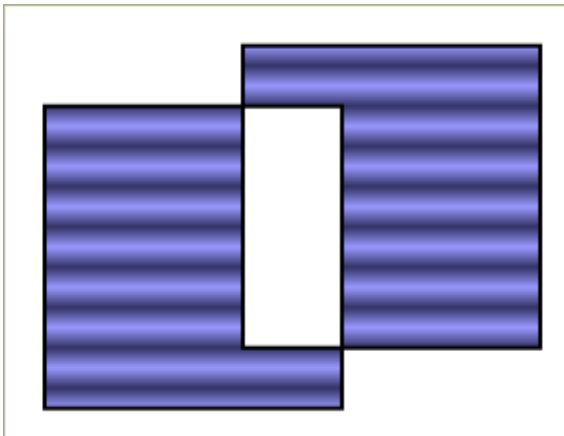
```

10 This markup is rendered as follows:



11

12 Without the Transform property, this markup would be rendered as follows:

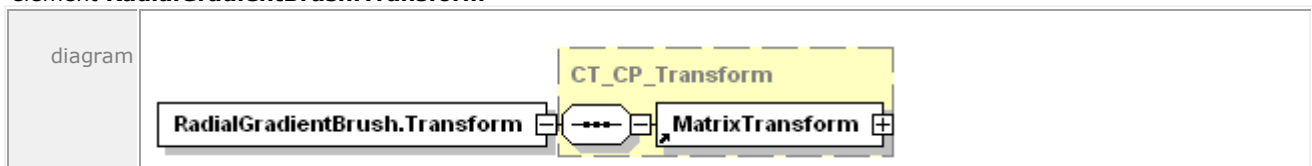


13

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*

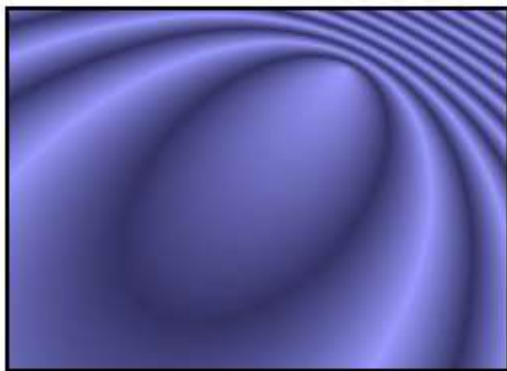
2 The following markup describes a rotation and reposition transform on a radial gradient:

```

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>

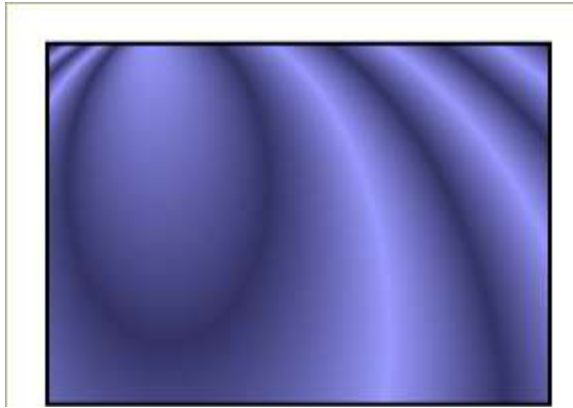
```

25 This markup is rendered as follows:



26

27 Without the Transform property, this markup is rendered as follows:



1

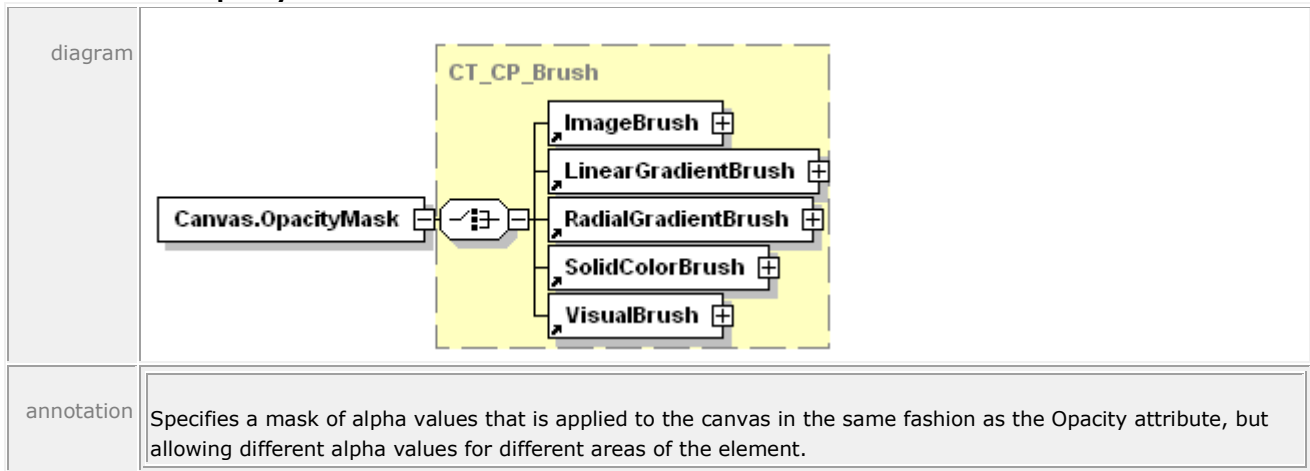
2 *end example]*

## 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"

```

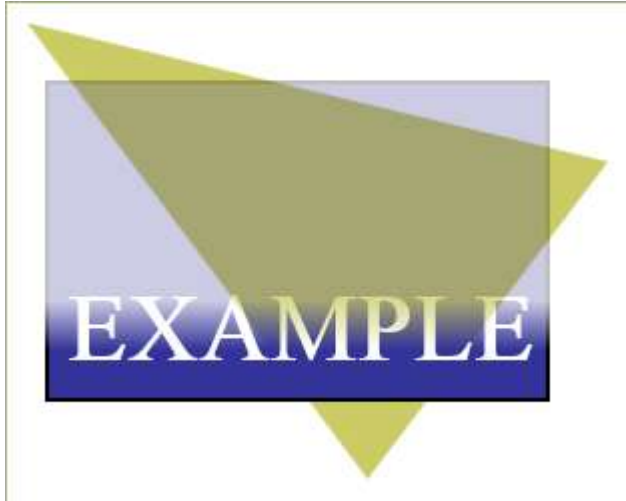


```

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**

diagram	
annotation	<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>

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"

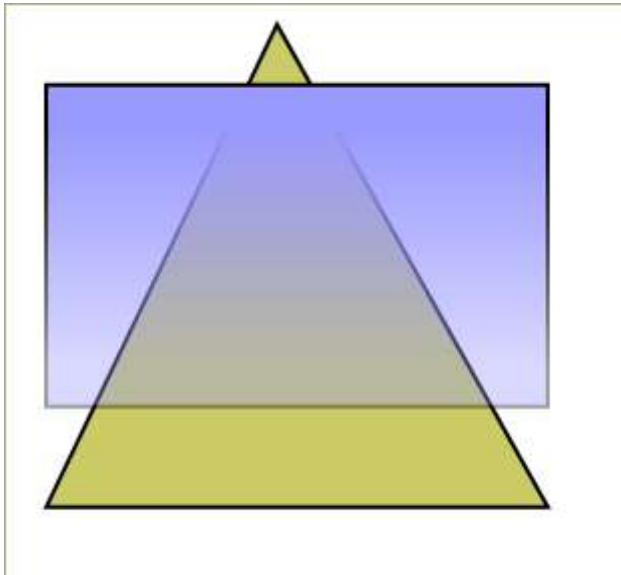
```

```

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:

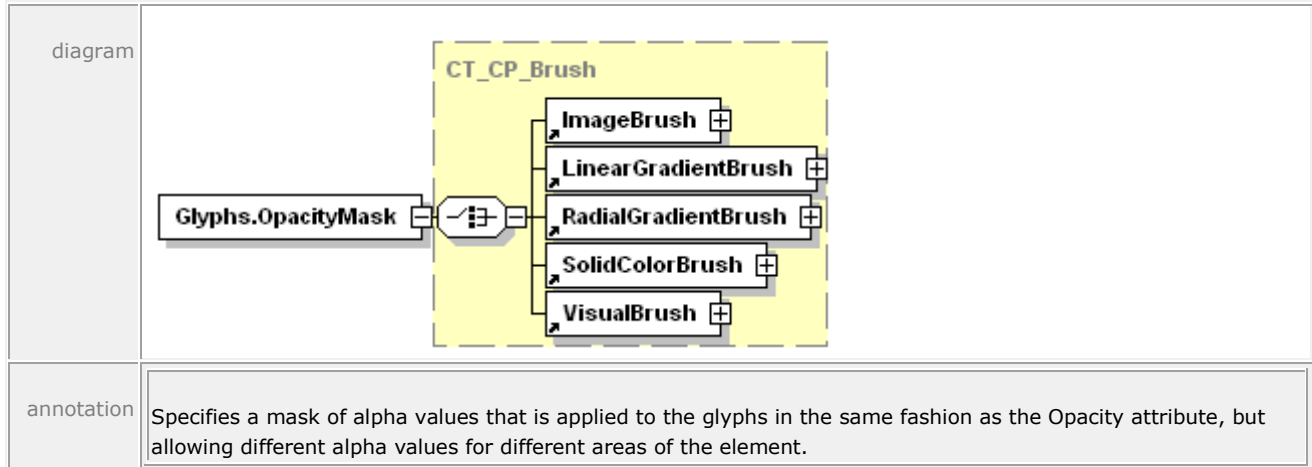


20

21 *end example]*

### 22 **14.5.3 <Glyphs.OpacityMask> Element**

23 element **Glyphs.OpacityMask**



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 OpenXPS  
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 OpenXPS Document and passing it through to printing  
7 consumers enables greater end-to-end color fidelity.

---

### 8 15.1 Color Support

9 OpenXPS Documents support sRGB and other color spaces, including scRGB, CMYK, N-Channel,  
10 and named colors. Consumers MUST support the following color features:

- 11 • Grayscale colors (single channel) in vector data, with and without alpha [M8.56]
- 12 • Grayscale colors in image data, using the JPEG, PNG, TIFF, or HDPhoto image formats  
13 [M8.57]
- 14 • sRGB colors (8 bit-per-channel) in vector data, with and without alpha [M8.1]
- 15 • sRGB colors in image data, using the JPEG, PNG, TIFF, or ~~Windows Media Photo~~[JPEG XR](#)  
16 image formats [M8.2]
- 17 • scRGB color specification in vector data, with and without alpha [M8.3]
- 18 • scRGB colors in image data, using the ~~Windows Media Photo~~[JPEG XR](#) image format  
19 [M8.4]
- 20 • CMYK colors in vector data [M8.5]
- 21 • CMYK colors in image data, using the TIFF or ~~Windows Media Photo~~[JPEG XR](#) image  
22 formats [M8.6]
- 23 • N-Channel colors in vector data [M8.7]
- 24 • N-Channel colors in image data, using the ~~Windows Media Photo~~[JPEG XR](#) image format  
25 [M8.8]

26 Producers and consumers MAY support the following color features:

- 27 • N-Channel colors in image data, using the TIFF image format [O8.19].

28 When non-sRGB color information is used, color value specifications are expressed using  
29 markup from the OpenXPS Document schema.

30 Consumers are not required to handle all color spaces natively through every processing stage,  
31 but, rather, MAY convert data specified in a color space other than sRGB to sRGB at an early  
32 stage [O8.1]. Consumers that do not handle natively colors other than sRGB can experience  
33 reduced fidelity.

34 The requirements and recommendations of this subclause and its subclauses pertain equally to  
35 raster and vector color content.

### 1 **15.1.1 sRGB Color Space**

2 The OpenXPS Document format supports colors in the sRGB color space for both vector and  
3 raster graphics.

### 4 **15.1.2 scRGB Color Space**

5 The OpenXPS Document format supports colors in the scRGB color space for both vector and  
6 raster graphics. Such scRGB colors are typically used without an ICC profile. The encoding of  
7 scRGB as specified in IEC 61966-2-2 does not include a gamut boundary definition, therefore  
8 the scRGB gamut boundary to be used for scRGB colors is implementation-defined.

### 9 **15.1.3 Gray Color Space**

10 Gray colors for vector elements can be specified as sRGB or scRGB colors with the red, blue and  
11 green components set to the same value. Alternatively, grayscale colors for vector elements  
12 can be specified as single channel monochrome with an associated ICC profile. Gray colors for  
13 raster images can be specified using any image format.

### 14 **15.1.4 CMYK Color Space**

15 CMYK color is supported through the use of color management transformations from an ICC  
16 profile.

### 17 **15.1.5 N-Channel Color Spaces**

18 N-channel color is supported through the use of color management transformations from an  
19 ICC profile.

### 20 **15.1.6 Named Color for Spot Colors and N-tone Images**

21 Named colors are supported through the use of color management transformations from an ICC  
22 profile.

### 23 **15.1.7 Identifying Output-Ready Color Spaces Using ICC Profiles**

24 An ICC profile, corresponding to a color space suitable for a particular device or device type,  
25 can be identified using a PrintTicket setting, as described in §9.1.9.

26 OpenXPS markup or profile embedding is used to identify such an ICC profile with the elements  
27 intended to be rendered in the native color space of the device or device-type.

28 If a consumer recognizes that a profile given in the syntax for a page element matches the  
29 PrintTicket output-ready ICC profile and that the PrintTicket output-ready ICC profile is suitable  
30 for the output device conditions, then the consumer SHOULD elect to treat the element colors  
31 as output-ready colors and not color-manage them, unless forced to do so for transparency  
32 effects or gradient blending [S8.21].

33 [*Note:* Elements using named colors in a workflow in which the consumer is expected to use the  
34 encoded name of a named color to lookup a device-specific color value are not affected by the  
35 use of the PrintTicket setting described here. *end note*]

### 36 **15.1.8 ICC Profiles**

37 OpenXPS Documents MAY include associated ICC profile parts [O2.3]. OpenXPS producers MAY  
38 include ICC profiles embedded in any image format (according to the restrictions of the image

1 file format) with any color space [O8.15]. For color spaces other than sRGB and scRGB,  
2 OpenXPS producers MUST provide color management using associated or embedded (for raster  
3 images) ICC profiles conforming to the requirements of the ICC Color Profile specification,  
4 ICC.1:2001-04 [M8.12]. OpenXPS producers MAY include ICC profiles for sRGB and scRGB  
5 color spaces [O8.16]. OpenXPS consumers MUST use associated and embedded ICC profiles,  
6 according to the precedence order of §15.3.8 for raster images and according to §15.2 for  
7 vector content [M8.53]. Optionally, OpenXPS producers and consumers MAY provide color  
8 management using ICC profiles conforming to the requirements of ISO 15076-1, "Image  
9 technology colour management — Architecture, profile format, and data structure — Part 1:  
10 Based on ICC.1:2004-10" [O8.9]. Producers MUST restrict associated ICC profiles to conform to  
11 the requirements of the older ICC Color Profile specification, ICC.1:2001-04, when consumer  
12 support of the newer ISO version cannot be ascertained [M8.58S8.15]. If a Producer includes  
13 an image with an embedded profile conforming to the requirements of ISO 15076-1, then the  
14 Producer MUST associate an ICC profile conforming to the requirements of the older ICC Color  
15 Profile specification, ICC.1:2001-04, to have precedence over such an embedded profile, when  
16 consumer support of the newer ISO version cannot be ascertained [M8.59]. All ICC profiles used  
17 in OpenXPS Documents MUST be one of the following [M8.13]:

- 18 • Input
- 19 • Output
- 20 • Monitor (RGB)
- 21 • ColorSpace Conversion
- 22 • Named Color

23 Supported profiles include Monochrome Input Profiles, Monochrome Display Profiles,  
24 Monochrome Output Profiles, Three-component Matrix-based Input Profiles, and RGB Display  
25 Profiles. The set of usable N-component LUT-based profiles is limited to 2-, 3-, 4-, 5-, 6-, 7-, or  
26 8-color channels. The set of usable Named Color profiles is limited to 1-, 2-, 3-, 4-, 5-, 6-, 7-,  
27 or 8-colors.

28 If consistency of appearance of grayscale images is important, the producer SHOULD adjust the  
29 gray tone response curve of such images before adding to the OpenXPS Document [S8.2].  
30 [*Note*: Some consumers do not correctly apply ICC profiles to grayscale images. *end note*]

31 An ICC profile MAY contain private tags [O8.17]. Implementations MAY act on private tags  
32 [O8.18] and MUST ignore and preserve private tags that they do not understand [M8.55].

---

## 33 **15.2 Vector Color Syntax**

34 This subclause describes specific considerations for including vector colors in OpenXPS  
35 Documents.

36 Vector colors can be specified in OpenXPS Document markup in the following locations:

- 37 • The Color attribute of the <SolidColorBrush> element
- 38 • The Color attribute of the <GradientStop> element
- 39 • The Fill attribute of the <Path> element
- 40 • The Fill attribute of the <Glyphs> element
- 41 • The Stroke attribute of the <Path> element

1 The last three locations are an abbreviated syntax for expressing a solid color brush with the  
2 specified color.

3 *Table 15-1. Syntax summary*

<b>Color type</b>	<b>Syntax</b>	<b>Example</b>
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 /swopcmypfile.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"	Color="ContextColor /namedtintprofile.icc 1.0, 1.0"
RGB with alpha	Color="ContextColor ProfileURI AlphaFloat, Chan0Float, Chan1Float, Chan2Float"	Color="ContextColor /RGBprofile.icc 1.0, 1.0, 1.0, 1.0"
Grayscale with alpha	Color="ContextColor ProfileURI AlphaFloat, Chan0Float"	Color="ContextColor /grayprofile.icc 1.0, 1.0"

4 Real numbers specified for color channel values of scRGB and ContextColor colors MUST NOT  
5 use exponent forms of numbers [M8.14].

6 Profiles associated as described in Table 15-1, and determined to be usable, MUST be used by  
7 consumers [M8.44].

8 It is the responsibility of consumers to determine profile usability. A profile associated as in  
9 Table 15-1 SHOULD be considered unusable by a consumer if

- 10 • The profile is not compatible with the context color syntax
- 11 • The profile contains optional tags that ambiguate OpenXPS use
- 12 • The profile contains invalid tag type signatures that invalidate OpenXPS use [S8.18].

13 In general, the presence of one or more optional tags in an ICC profile does not make the  
14 profile unusable. A consumer incapable of supporting a particular ICC profile tag that is optional  
15 in both ICC and OpenXPS MAY treat this tag as a user-defined custom tag, and therefore ignore  
16 it [O8.13].

17 If no usable profile is present in a context color syntax, then a consumer MUST apply a color  
18 rule based on the context color syntax [M8.45]. The context color value(s) are interpreted to be  
19 the encoding of a particular color space as follows:



- 1 • Single component integer default for vector data MUST be grayscale with the sRGB non-  
2 linearity, black point, and white point [M8.46].
- 3 • Three component integer default for vector data MUST be sRGB [M8.47].
- 4 • Three component float default for vector data MUST be scRGB [M8.48].
- 5 • The specific CMYK to be used as the four component data default for vector data MUST  
6 be determined by the consumer [M8.49].
- 7 • N-Channel data with  $N \leq 3$  and any named color data: the data of the first channel  
8 MUST be interpreted independently as grayscale [M8.50]. Other channels are  
9 disregarded.
- 10 • N-Channel with  $N > 4$  MUST be treated as four component data using the four  
11 component data default for vector data determined by the consumer [M8.51].

12 When no usable profile is present a consumer MAY choose to instantiate an error condition  
13 [O8.14].

14 A producer MUST associate or embed a usable color profile if the color rules above do not  
15 guarantee appropriate color interpretation for the vector color content [M8.52].

### 16 **15.2.1 sRGB Color Syntax**

17 The sRGB color syntax is the same as that used in HTML, with the red, green, and blue  
18 channels represented by two hexadecimal digits. OpenXPS Documents can specify an sRGB  
19 color either with or without an alpha channel value, which is also expressed as two hexadecimal  
20 digits.

21 The syntax is as follows (without alpha):

22 #RRGGBB

23 or (with alpha):

24 #AARRGGBB

25 When an sRGB color is specified without an alpha value, an alpha of "FF" is implied.

### 26 **15.2.2 scRGB Color Syntax**

27 The scRGB color syntax allows OpenXPS Document producers to specify a color using the full  
28 scRGB color space, which is much larger than the sRGB color space and can represent the  
29 entire range 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 Grayscale syntax**

10 OpenXPS Document producers specify grayscale colors using the context color syntax, which  
11 allows specification of a monochrome 'GRAY' ICC profile and an individual color channel value  
12 as a real number. The context color MUST specify the matching number of channel float values  
13 [M8.17].

14 The syntax is as follows:

15 `ContextColor ProfileURI AlphaFloat, Chan0Float`

16 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST  
17 be added as a Required Resource relationship to the FixedPage part [M2.10].

18 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be  
19 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.16]. Channel float  
20 values MUST also be clamped to the valid range from 0.0 to 1.0 before further processing.  
21 Before the value is used as input for an ICC profile color transformation, it MUST be linearly  
22 scaled [with specified rounding/clipping] to the range from 0 to 255 or from 0 to 65535,  
23 depending on whether the profile uses 8-bit or 16-bit input tables [M8.31].

### 24 **15.2.4 CMYK Color Syntax**

25 OpenXPS Document producers specify CMYK colors using the context color syntax, which allows  
26 specification of an ICC profile and the individual color channel values as real numbers.

27 The syntax is as follows:

28 `ContextColor ProfileURI AlphaFloat, Chan0Float, Chan1Float, Chan2Float,`  
29 `Chan3Float`

30 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST  
31 be added as a Required Resource relationship to the FixedPage part [M2.10].

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 [M8.16]. Channel float  
34 values MUST also be clamped to the valid range from 0.0 to 1.0 before further processing.  
35 Before the value is used as input for an ICC profile color transformation, it MUST be linearly  
36 scaled (with specified rounding/clipping) to the range from 0 to 255 or from 0 to 65535,  
37 depending on whether the profile uses 8-bit or 16-bit input tables [M8.31].

### 1 **15.2.5 N-Channel Color Syntax**

2 OpenXPS Document producers specify N-channel colors using the context color syntax, which  
3 allows specification of an ICC profile and the individual color channel values as real numbers.  
4 The syntax is expressed as follows:

5 `ContextColor ProfileURI AlphaFloat, Chan0Float, ..., ChanN-1Float`

6 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST  
7 be added as a Required Resource relationship to the FixedPage part [M2.10].

8 The profile can be a 2-, 3-, 4-, 5-, 6-, 7- or 8-channel N-clr profile (indicated by using one of  
9 the {'2CLR' ... '8CLR'} values in the profile header color space signature field). The context color  
10 MUST specify the matching number of channel float values [M8.17].

11 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be  
12 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.18]. Channel float  
13 values MUST also be clamped to the valid range from 0.0 to 1.0 before further processing.  
14 Before the value is used as input for an ICC profile color transformation, it MUST be linearly  
15 scaled (with specified rounding/clipping) to the range from 0 to 255 or from 0 to 65535,  
16 depending on whether the profile uses 8-bit or 16-bit input tables [M8.31].

17 [*Example*: For duotone 2-clr content (with color-managed color mixing) the syntax is:

18 `ContextColor ProfileURI AlphaFloat, Chan0Float, Chan1Float`

19 *end example*]

20 For 1-channel color, i.e., monochrome, use a monochrome input or output profile (profile  
21 header color space signature is 'GRAY'). The profile MUST include the ICC-optional AToB1Tag  
22 (relative colorimetric intent) if the single color is chromatic (not neutral) [M8.32]. [*Example*:

23 `ContextColor ProfileURI AlphaFloat, Chan0Float`

24 *end example*]

25 If the OpenXPS system environment allows the use of ICC ISO 15076-1 profiles, the optional  
26 colorantTableTag SHOULD be included in such ISO 15076-1 profiles to indicate the names and  
27 corresponding PCS values of the individual N-color colorants [S8.16]. (See §15.1.8 for the  
28 appropriate use of ISO 15076-1 profiles.)

### 15.2.6 Named Color Syntax

A *named color* is an industry-defined color specification that identifies a particular color in a well-defined color system, usually for the purpose of printing. There are currently several named color systems. In OpenXPS, a named color is expressed as a combination of an ink name and transform information stored in an ICC profile and a tint level (percentage ink dilution) given in the OpenXPS context color syntax. The OpenXPS context color syntax allows specification of one or more named color tint values and association of an ICC profile. The syntax is expressed as follows:

```
ContextColor ProfileURI AlphaFloat,Tint0Float,...,TintN-1Float
```

Two ICC profile approaches are available for named colors, one using ICC monochrome profiles that each include a tint LUT for a single named color, and the other using ICC Named Color type profiles that each can include 100% color values for 1, 2, 3, 4, 5, 6, 7, or 8 named colors. In both cases, the OpenXPS context color syntax MUST specify the matching number of tint float values [M8.55].

A named color with an associated tint LUT MUST be implemented in an OpenXPS Document using an associated ICC monochrome profile [M8.33]. In this case, the ICC profile MUST contain the tint LUT for a single named color [M8.34]. The ICC profile MUST be an ICC monochrome input or output profile [M8.35]. The profile header color space signature MUST be 'GRAY' [M8.37]. The profile MUST include an AtoB1Tag (relative colorimetric rendering intent), mapping the named color tint values to valid PCS values [M8.19], in addition to the ICC-required grayTRCTag (not used for OpenXPS named colors). The ASCII prefix-root-suffix name of the named color MUST be encoded into the profileDescriptionTag of the ICC profile [M8.36] so that a consumer MAY use the profile to obtain the encoded name of the named color [O8.20]. A consumer MAY use the encoded name of a named color to lookup a device-specific color value for the named color [O8.21].

The context color syntax for referencing a single named color is as follows:

```
ContextColor ProfileURI AlphaFloat,TintFloat
```

A single named color MAY be implemented in an OpenXPS Document using an associated ICC Named Color type profile [O8.11]. Two or more named colors implemented in an OpenXPS Document using a single associated profile MUST use an ICC Named Color type profile [M8.38]. An ICC Named Color type profile MUST contain the namedColor2Tag including the ASCII prefix-root-suffix name for each named color [M8.39], so that a consumer MAY use the profile to obtain the encoded name of the named color [O8.22]. The namedColor2Tag MUST be populated with the ICC PCS color value for each named color [M8.40] and MAY be populated with specific device color values for each named color [O8.12]. A named color duotone, tritone, etc., can be implemented in this way. A consumer MAY use the encoded name of a named color to lookup a device-specific color value for the named color [O8.23].

[*Example*: For duotone named color content (with NO color managed color mixing) the syntax is:

```
ContextColor ProfileURI AlphaFloat,Tint0Float,Tint1Float
```

*end example*]

1 `ProfileURI` specifies a part containing the binary data of the color profile. The profile URI MUST  
 2 be added as a Required Resource relationship to the FixedPage part [M2.10]. `AlphaFloat`  
 3 specifies the alpha to be applied to the named color. `TintFloat` specifies how diluted with  
 4 respect to the color system's white color point the named color is, with 1.0 being the pure  
 5 named color and 0.0 being fully diluted.

6 Although alpha values smaller than 0.0 and larger than 1.0 can be specified, they MUST be  
 7 clamped to the valid range from 0.0 to 1.0 before any further processing [M8.20]. The tint float  
 8 value MUST also be clamped to the valid range from 0.0 to 1.0 before further processing.  
 9 Before the value is used as input for an ICC profile color transformation, it MUST be linearly  
 10 scaled (with specified rounding/clipping) to the range from 0 to 255 or from 0 to 65535,  
 11 depending on whether the profile uses 8-bit or 16-bit input tables [M8.31].

12 Consumers MAY use the ASCII name in the ICC profile or MAY compute a color approximation  
 13 using the specified color value in the ICC profile. When a named color is used in a gradient  
 14 brush or with transparency, the results of these two methods MAY differ significantly [O8.3].

---

## 15 15.3 Colors in Raster Images

16 This subclause describes specific considerations for including color raster images in OpenXPS  
 17 Documents.

### 18 15.3.1 sRGB Raster Images

19 OpenXPS Documents support sRGB raster images in the following formats:

- 20 • JPEG
- 21 • PNG
- 22 • TIFF
- 23 • ~~Windows Media Photo~~JPEG XR

24 The following ~~Windows Media Photo~~JPEG XR pixel format mnemonics are supported:

- 25 • ~~WICPixelFormat~~24bppRGB
- 26 • ~~WICPixelFormat~~24bppBGR
- 27 • ~~WICPixelFormat~~32bppBGR
- 28 • ~~WICPixelFormat~~32bppBGRA
- 29 • ~~WICPixelFormat~~32bppPBGRA
- 30 • ~~WICPixelFormat~~48bppRGB
- 31 • ~~WICPixelFormat~~64bppRGBA
- 32 • ~~WICPixelFormat~~64bppPRGBA

33 Pixel formats ~~WICPixelFormat~~32bppPBGRA and ~~WICPixelFormat~~64bppPRGBA are pre-multiplied  
 34 alpha formats. See §18.4.1 for details.

35 The following ~~Windows Media Photo~~JPEG XR packed pixel format mnemonics are supported:

- 36 • ~~WICPixelFormat~~16bppBGR555
- 37 • ~~WICPixelFormat~~16bppBGR565

- ~~WICPixelFormat32bppBGR101010~~

See §9.1.5 for more details.

### 15.3.2 scRGB Raster Images

OpenXPS Documents support scRGB raster images only in the ~~Windows Media Photo~~[JPEG XR](#) image format. The following pixel format [mnemonics](#) are supported:

- ~~WICPixelFormat48bppRGBFixedPoint~~
- ~~WICPixelFormat48bppRGBHalf~~
- ~~WICPixelFormat96bppRGBFixedPoint~~
- ~~WICPixelFormat128bppRGBFloat~~
- ~~WICPixelFormat64bppRGBAFixedPoint~~
- ~~WICPixelFormat64bppRGBFixedPoint~~
- ~~WICPixelFormat64bppRGBHalf~~
- ~~WICPixelFormat64bppRGBHalf~~
- ~~WICPixelFormat128bppRGBAFixedPoint~~
- ~~WICPixelFormat128bppRGBFixedPoint~~
- ~~WICPixelFormat128bppRGBAFixedPoint~~
- ~~WICPixelFormat128bppRGBFloat~~
- ~~WICPixelFormat128bppPRGBAFixedPoint~~
- ~~WICPixelFormat32bppRGBE~~

Pixel format ~~WICPixelFormat128bppPRGBAFixedPoint~~ is a pre-multiplied alpha format. See §18.4.1 for details.

### 15.3.3 Gray Raster Images

OpenXPS Documents support gray raster images in the following formats:

- JPEG
- PNG
- TIFF
- ~~Windows Media Photo~~[JPEG XR](#)

The following ~~Windows Media Photo~~[JPEG XR](#) pixel format [mnemonics](#) are supported:

- ~~WICPixelFormatBlackWhite~~
- ~~WICPixelFormat8bppGray~~
- ~~WICPixelFormat16bppGray~~
- ~~WICPixelFormat16bppGrayFixedPoint (scRGB range)~~
- ~~WICPixelFormat16bppGrayHalf (scRGB range)~~
- ~~WICPixelFormat32bppGrayFixedPoint (scRGB range)~~
- ~~WICPixelFormat32bppGrayFloat~~

### 1 **15.3.4 CMYK Raster Images**

2 CMYK images are stored in TIFF or ~~Windows Media Photo~~[JPEG XR](#) format.

#### 3 **15.3.4.1 TIFF CMYK Raster Images**

4 CMYK TIFF image tags are described in §9.1.5.3.

5 ICC profiles can be associated with CMYK raster images by using an ICC profile embedded in  
6 the TIFF file (tag 34675) or associated using the mechanism described in §15.3.8.

#### 7 **15.3.4.2 ~~Windows Media Photo~~[JPEG XR](#) CMYK Raster Images**

8 The ~~Windows Media Photo~~[JPEG XR](#) CMYK format is described in the ~~Windows Media~~  
9 ~~Photo~~[JPEG XR](#) specification. The following format [mnemonics](#) are supported:

- 10 • ~~WICPixelFormat~~32bppCMYK
- 11 • ~~WICPixelFormat~~40bppCMYKAlpha
- 12 • ~~WICPixelFormat~~64bppCMYK
- 13 • ~~WICPixelFormat~~80bppCMYKAlpha

14 [\[Note: The following JPEG XR pixel formats mnemonics are not supported:](#)

- 15 • [32bppCMYKDIRECT](#)
- 16 • [64bppCMYKDIRECT](#)
- 17 • [40bppCMYKDIRECTAlpha](#)
- 18 • [80bppCMYKDIRECTAlpha](#)

19 [end note\]](#)

#### 20 **15.3.4.3 JPEG CMYK Raster Images**

21 Support for JPEG CMYK images varies by implementation and SHOULD NOT be used in  
22 OpenXPS Documents [S2.7]. See §9.1.5.1 for more details.

### 23 **15.3.5 N-channel Raster Images**

24 N-channel images are stored in the ~~Windows Media Photo~~[JPEG XR](#) image file format using an  
25 ICC profile. The following format [mnemonics](#) are supported:

- 26 • ~~WICPixelFormat~~24bpp3Channels
- 27 • ~~WICPixelFormat~~48bpp3Channels
- 28 • ~~WICPixelFormat~~32bpp4Channels
- 29 • ~~WICPixelFormat~~64bpp4Channels
- 30 • ~~WICPixelFormat~~40bpp5Channels
- 31 • ~~WICPixelFormat~~80bpp5Channels
- 32 • ~~WICPixelFormat~~48bpp6Channels
- 33 • ~~WICPixelFormat~~96bpp6Channels
- 34 • ~~WICPixelFormat~~56bpp7Channels
- 35 • ~~WICPixelFormat~~112bpp7Channels

- 1 • [\\_WICPixelFormat64bpp8Channels](#)
- 2 • ~~WICPixelFormat128bpp8Channels~~
- 3 • [\\_WICPixelFormat32bpp3ChannelsAlpha](#)
- 4 • ~~WICPixelFormat64bpp3ChannelsAlpha~~
- 5 • [\\_WICPixelFormat40bpp4ChannelsAlpha](#)
- 6 • ~~WICPixelFormat80bpp4ChannelsAlpha~~
- 7 • [\\_WICPixelFormat48bpp5ChannelsAlpha](#)
- 8 • ~~WICPixelFormat96bpp5ChannelsAlpha~~
- 9 • [\\_WICPixelFormat56bpp6ChannelsAlpha](#)
- 10 • ~~WICPixelFormat112bpp6ChannelsAlpha~~
- 11 • [\\_WICPixelFormat64bpp7ChannelsAlpha](#)
- 12 • ~~WICPixelFormat128bpp7ChannelsAlpha~~
- 13 • [\\_WICPixelFormat72bpp8ChannelsAlpha](#)
- 14 • ~~WICPixelFormat144bpp8ChannelsAlpha~~

15 The profile can be a 2-, 3-, 4-, 5-, 6-, 7- or 8-channel N-clr profile (indicated by using one of  
16 the {'2CLR' ... '8CLR'} values in the profile header color space signature field).

17 For 1-channel color, i.e., monochrome, use a monochrome input (or output) profile. The profile  
18 MUST include the ICC-optional AToB1Tag (relative colorimetric intent) if the single color is  
19 chromatic (not neutral) [M8.32].

20 If the OpenXPS system environment allows the use of ICC ISO 15076-1 profiles, the optional  
21 colorantTableTag SHOULD be included in such ISO 15076-1 profiles to indicate the names and  
22 corresponding PCS values of the individual N-color colorants [S8.16]. (See §15.1.8 for the  
23 appropriate use of ISO 15076-1 profiles.)

### 24 **15.3.6 Named Color Raster Images**

25 A *named color* is an industry-defined color specification that identifies a particular color in a  
26 well-defined color system, usually for the purpose of printing. There are currently several  
27 named color systems. In OpenXPS, a named color is expressed as a combination of an ink  
28 name and transform information stored in an ICC profile.

29 Named color raster images are stored in the ~~Windows Media Photo~~[JPEG XR](#) image file format  
30 using an ICC profile that maps the tint channel combinations to valid PCS values. See §15.3.5  
31 for pixel format definitions.

32 Consumers unaware of named colors can then compute color approximations using the PCS  
33 values computed from the profile.

34 Two ICC profile approaches are available for named colors, one using ICC monochrome profiles  
35 that each include a tint LUT for a single named color, and the other using ICC Named Color type  
36 profiles that each can include 100% color values for 1, 2, 3, 4, 5, 6, 7, or 8 named colors.

37 A monochrome named color raster image can have a tint LUT encoded in an ICC monochrome  
38 input or output profile. The profile header color space signature is 'GRAY'. The profile includes  
39 an AtoB1Tag (relative colorimetric rendering intent), for the tint LUT mapping the named color  
40 tint values to valid PCS values. The ASCII prefix-root-suffix name of the named color is



1 encoded into the profileDescriptionTag of the ICC profile so that a consumer MAY use the profile  
 2 to obtain the encoded name of the named color [O8.22]. A consumer MAY use the encoded  
 3 name of a named color to lookup a device-specific color value for the named color [O8.23].

4 A multi-tone named color raster image can have an ICC Named Color type profile. An ICC  
 5 Named Color type profile MUST contain the namedColor2Tag including the ASCII prefix-root-  
 6 suffix name for each named color [M8.39] so that a consumer MAY use the profile to obtain the  
 7 encoded name of the named color [O8.22]. The namedColor2Tag MUST be populated with the  
 8 ICC PCS color value for each named color [M8.40] and MAY be populated with specific device  
 9 color values for each named color [O8.12]. A named color duotone, tritone, etc., can be  
 10 implemented in this way. A consumer MAY use the encoded name of a named color to lookup a  
 11 device-specific color value for the named color [O8.23].

12 Consumers MAY use the ASCII name in the ICC profile or MAY compute a color approximation  
 13 using a specified color value in the ICC profile; the results of these two methods MAY differ  
 14 significantly [O8.25].

### 15 **15.3.7 Device Color Raster Images**

16 Device color (N-channel) raster images are stored in the ~~Windows Media Photo~~JPEG XR image  
 17 file format in the same manner as a named color raster image. See§15.1.7 for more details.  
 18 RGB and CMYK raster images can also be stored in the TIFF image file format. JPEG CMYK  
 19 images SHOULD NOT be used [S2.7].

### 20 **15.3.8 Images and Color Profile Association**

21 Images can depend on color profiles using one of two methods:

- 22 • Associated: Color profile contained in a separate part associated with the image.
- 23 • Embedded: Color profile embedded in an image using the image format specific mechanism.

24 When associating a profile with an image the syntax for the ImageSource attribute is as  
 25 follows:

```
26 {ColorConvertedBitmap ImageSourceURI ProfileURI}
```

27 ImageSourceURI Specifies the URI of an image resource. The image URI MUST be added as a  
 28 Required Resource relationship to the FixedPage part [M2.10].

29 ProfileURI specifies a part containing the binary data of the color profile. The profile URI MUST  
 30 be added as a Required Resource relationship to the FixedPage part [M2.10].

31 *[Example:*

```
32 <ImageBrush ImageSource="{ColorConvertedBitmap ../Resources/Images/image.tif  

  33 ../Metadata/profile.icc}" ... />
```

34 *end example]*

35 It is the responsibility of consumers to determine the usability of embedded or associated  
 36 profiles. A profile associated or embedded with an image SHOULD be considered unusable by a  
 37 consumer if

- 38 • The profile is not compatible with the pixel format of the image
- 39 • The profile contains optional tags that ambiguate OpenXPS use
- 40 • The profile contains invalid tag type signatures that invalidate OpenXPS use [S8.17].

1 In general, the presence of one or more optional tags in an ICC profile does not make the  
 2 profile unusable. A consumer incapable of supporting a particular ICC profile tag that is optional  
 3 in both ICC and OpenXPS MAY treat this tag as a user-defined custom tag, and therefore ignore  
 4 it [O8.13].

5 If present and usable, an associated profile MUST be used by consumers [M8.41]. A usable  
 6 associated color profile overrides an embedded color profile and is processed instead of any  
 7 embedded color profile.

8 If present and usable, a color profile embedded in an image file MUST be used by consumers  
 9 when no usable associated profile is present with the image [M8.42].

10 If no usable profile is present with an image, then a consumer MUST apply a color rule based  
 11 on the pixel format. Each pixel format is interpreted to be the encoding of a particular color  
 12 space as shown in Table 15–1 [M8.30].

13 When no usable profile is present a consumer MAY choose to instantiate an error condition  
 14 [O8.14].

15 A producer MUST associate or embed a usable color profile if the color rules of Table 15–1 do  
 16 not guarantee appropriate color interpretation for an image [M8.43].

17 *Table 15–2. Color Space Pixel Format Defaults*

<b>Pixel Formats</b>	<b>Color Space</b>
Integer 1-Channel	Grayscale using non-linearity, black point, and white point from sRGB
Fixed Point 1-Channel	
Half-Float 1-Channel	
Floating Point 1-Channel	
Integer 3-Channel	sRGB
Floating Point 3-Channel	scRGB
Half-Float 3-Channel	
Fixed-Point 3-Channel	
Integer 4-Channel	CMYK
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)	

18 The sRGB non-linearity, white point, and black point can be applied to single channel grayscale  
 19 data using the equations of IEC 61966-2-1, §5.2, by setting R=G=B equal to the grayscale  
 20 value.

21 The specific CMYK to be used as the four-component raster data default, and the N-Channel  
 22 (N=>4) default, is implementation-defined. In the absence of specific requirements the use of  
 23 CGATS/SWOP TR003 2007 CMYK is RECOMMENDED [S8.19]. Alternatively, a consumer MAY  
 24 choose to instantiate an error condition [O8.14].

1 [Note: A profile for CGATS/SWOP TR003 2007 CMYK is available from the ICC Profile Registry,  
2 specifically SWOP2006\_Coated3v2.icc. end note]

---

### 3 **15.4 Registration Marks for Color Separations**

4 Producers MAY elect to generate content that provides registration marks for consumers that  
5 perform color separation [O8.5].

6 The named color syntax can be used for registration marks that are intended to be rendered on  
7 every separation. A document registration named color can be identified at the document level  
8 using a PrintTicket setting (see §9.1.9).

9 A document registration named color identified in a PrintTicket MAY occur in an OpenXPS  
10 Document using the single named color and monochrome profile with tint LUT syntax  
11 (see §15.2.6) [O8.26]. The name of the document registration named color is given in the  
12 profile's profileDescriptionTag according to §15.2.6. Such a document registration named color  
13 SHOULD be unique for that use in the OpenXPS Document instance[S8.22].

14 For consumers that do not perform separation, the document registration named color ICC  
15 profile is used to compute output colorant values corresponding to the document registration  
16 named color. For consumers that do perform separation, the occurrence of the document  
17 registration named color in a color syntax is *only* an indicator that the tint level supplied in the  
18 syntax SHOULD be used when drawing the registration marking in each colorant separation  
19 [S8.7]. Producers SHOULD create the profile for the document registration named color in such  
20 a way that it does not lay down excessive ink when printed on a device that does not perform  
21 separation [S8.8].

---

### 22 **15.5 Alpha and Gradient Blending**

23 For consumers that handle colors other than sRGB, it is necessary to understand how they can  
24 be blended to create gradient or transparency effects. A page-level PrintTicket setting can be  
25 used to specify the blending color space that SHOULD be used for blending gradients and  
26 transparencies [S8.9] (see §9.1.9).

27 If a consumer understands the blending color space PrintTicket setting, it SHOULD convert all  
28 color to the specified blending color space before performing a blend operation [S8.9]. For  
29 gradients, the specified blending color space is used only if no gradient stop color values are  
30 specified using sRGB or scRGB colors. If any of the gradient stop color values are specified  
31 using sRGB or scRGB colors or the consumer does not understand the blending color space  
32 PrintTicket setting, the color interpolation mode of the gradient brush MUST be used instead  
33 [M8.25].

34 Consumers MUST support alpha and gradient blending in sRGB [M8.1], but they MAY support  
35 alpha and gradient blending with other color spaces such as scRGB or CMYK [O8.6]. The  
36 behavior of documents using non-sRGB alpha and gradient blending is implementation-specific.  
37 Consumers that encounter any document using non-sRGB colors MAY process those colors  
38 using conversion to the simpler sRGB color space, resulting in deviations, especially for alpha  
39 blending [O8.6].

## 1 15.6 Color Rendering Intent

2 ICC profiles contain multiple color transformation options, identified in the ICC Color Profiles  
3 specification as ICC rendering intents. For color elements that are to be color managed, a page  
4 level default color rendering intent, can be identified using a PrintTicket (see §9.1.9).

5 In the absence of such information, in a typical case, with ICC profiles conforming to the ICC  
6 Color Profile specification, ICC.1:2001-04 [M8.12], a consumer SHOULD apply the defaults  
7 shown in Table 15-3 ~~Table 15.4~~ [S8.20].

8 *Table 15-3. Recommended ICC rendering intent usage*

<b>Color type</b>	<b>Object type</b>	<b>ICC Source Rendering Intent</b>	<b>ICC Destination Rendering Intent</b>
sRGB	Raster image	Perceptual	Perceptual
sRGB	Vector	Relative Colorimetric	Relative Colorimetric*
scRGB	Raster image	Perceptual	Perceptual
scRGB	Vector	Relative Colorimetric	Relative Colorimetric*
RGB color space	Raster image	Perceptual	Perceptual
RGB color space	Vector	Relative Colorimetric	Relative Colorimetric*
CMYK, Gray color space	Raster image	Relative Colorimetric	Perceptual
CMYK, Gray color space	Vector	Relative Colorimetric	Relative Colorimetric*
Named color, N-Channel	Any	Relative Colorimetric	Relative Colorimetric

9 \*In the optional case, with ICC profiles conforming to the requirements of ISO 15076-1, based  
10 on ICC.1:2004-10 [O8.9], the Saturation Rendering Intent components of the profiles should  
11 be optimized for business graphics and may provide preferred results.

## 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 OpenXPS 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 OpenXPS 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 OpenXPS 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].

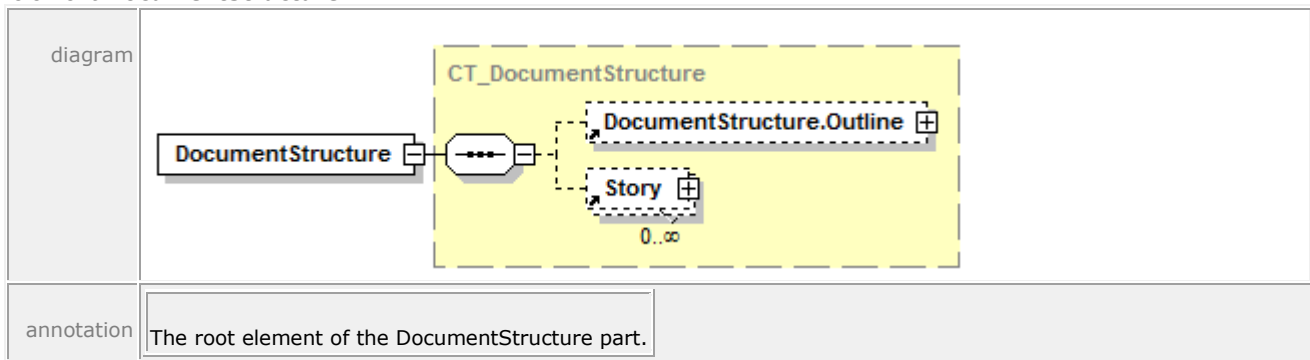
1 Document structure markup SHOULD NOT refer to a single named element more than once in  
 2 the document content or to a named element that embeds another named element that it also  
 3 refers to. When referring to a <Canvas> element, producers SHOULD consider all descendant  
 4 elements to be referenced in markup order [S9.3]. Consumers MAY choose to interpret these  
 5 scenarios as duplicate document content [O9.3].

6 Children of <VisualBrush> elements SHOULD NOT be referenced by document structure  
 7 markup [S9.30].

8 Because each named element in a FixedPage part that is intended as an addressable location is  
 9 specified in the <PageContent.LinkTargets> element in the FixedDocument part, consumers  
 10 MAY first attempt to locate named elements directly from the FixedDocument part [O9.4].

#### 11 16.1.1.1 <DocumentStructure> Element

12 element **DocumentStructure**



13 The <DocumentStructure> element is the root element of the DocumentStructure part. That  
 14 element MAY contain a single <DocumentStructure.Outline> element and zero or more  
 15 <Story> elements [O9.14].

16 *Example 16–1. Document structure markup*

```

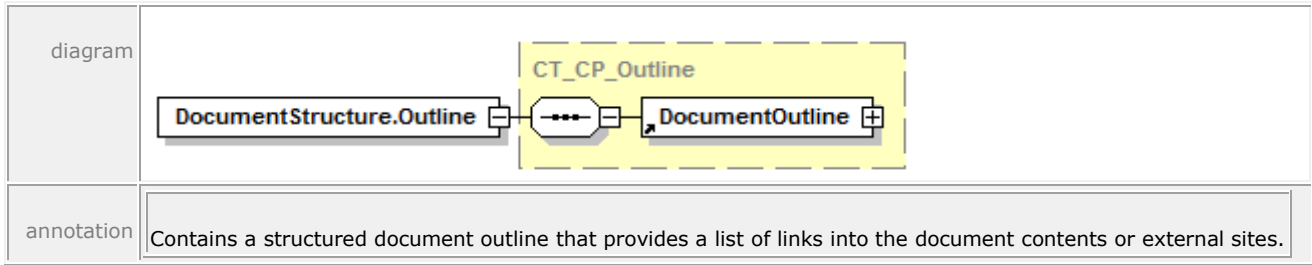
17 <DocumentStructure
18
19 | xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
20 | mentstructure">
21 |   <DocumentStructure.Outline>
22 |     ...
23 |   </DocumentStructure.Outline>
24 |   <Story>
25 |     ...
26 |   </Story>
27 |   <Story>
28 |     ...
29 |   </Story>
30 | </DocumentStructure>

```

31 *end example]*

1 **16.1.1.2 <DocumentStructure.Outline> Element**

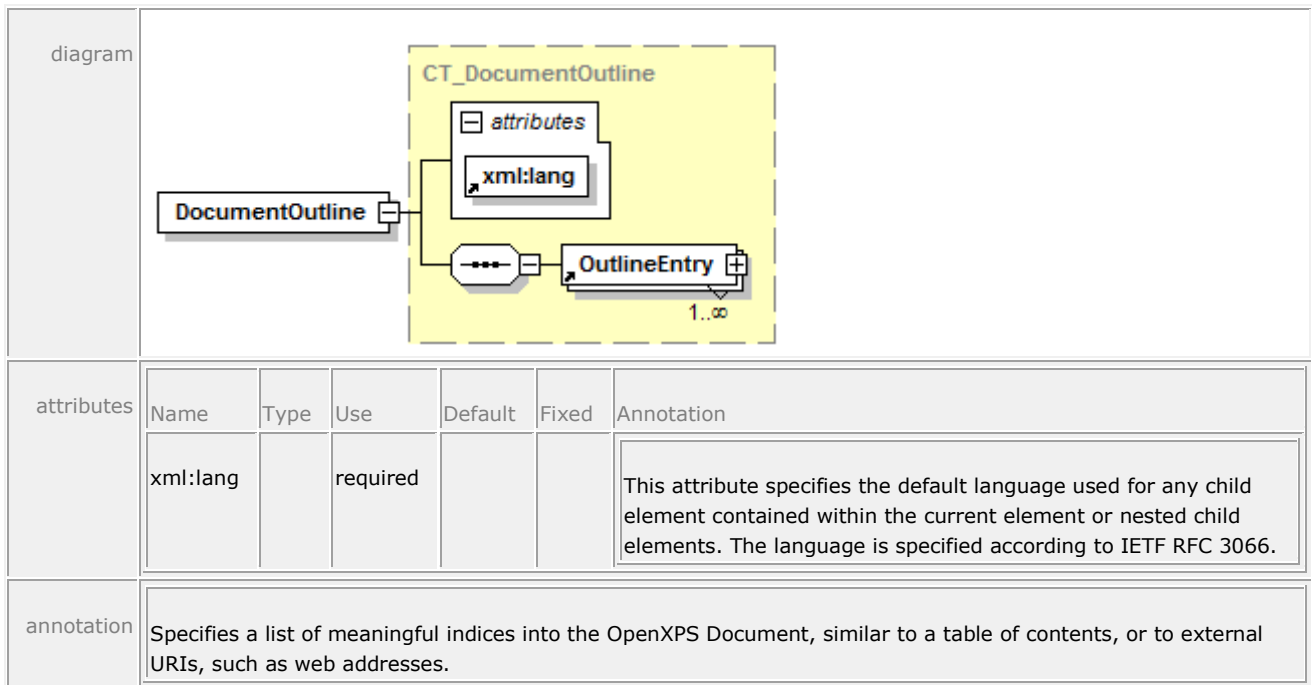
2 element **DocumentStructure.Outline**



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><u>ST_IntGEOne</u></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	<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.
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.																										
<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
11 |   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
12 |   mentstructure">
13 |     <DocumentStructure.Outline>
14 |       <DocumentOutline>
15 |         <OutlineEntry
```



```

1         OutlineLevel="1"
2         Description="1. Documents"
3         OutlineTarget=" ../FixedDoc.fdoc#Documents_1" />
4     <OutlineEntry
5         OutlineLevel="2"
6         Description="1.1. Paragraphs"
7         OutlineTarget=" ../FixedDoc.fdoc#Paragraphs_1_1" />
8     </DocumentOutline>
9 </DocumentStructure.Outline>
10 </DocumentStructure>

```

11 A consumer might display this information as follows, with the first entry linked to Documents\_1  
12 and the second entry linked to Paragraphs\_1\_1.

- 13 1. Documents
- 14 1.1. Paragraphs

15 *end example]*

16 **16.1.1.5 <Story> Element**

17 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 OpenXPS Document.					

18 The <Story> element is the root for a single story and orders all of the story fragments  
19 containing content structure information such as sections, paragraphs, and tables. Each story  
20 has a unique name that is used to correlate the content structure for each page to that story.  
21 The <Story> element contains one or more <StoryFragmentReference> elements.

1 **16.1.1.6 <StoryFragmentReference> Element**

2 element **StoryFragmentReference**

diagram						
attributes	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	<a href="#">ST_IntGEOne</a>	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	Identifies the StoryFragments part where this individual story fragment is defined.					

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 OpenXPS Document each story fragment appears, consumers can easily access only the pages  
 6 that 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 OpenXPS Document referenced within the fixed document sequence and fixed  
 9 document markup, starting at the fixed payload root. This value is identified in the Page  
 10 attribute. The StoryFragments part containing the corresponding content structure is referenced  
 11 by retrieving the part associated via relationship from the indicated page. This allows  
 12 consumers to access only the pages of the document that contain the story of interest. It is also  
 13 possible for a single 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 [M9.11].

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
7 xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
8 mentstructure">
9 <Story StoryName="Story1">
10 <StoryFragmentReference Page="1"/>
11 <StoryFragmentReference Page="2"/>
12 <StoryFragmentReference Page="4"/>
13 </Story>
14 <Story StoryName="Story2">
15 <StoryFragmentReference Page="2"/>
16 <StoryFragmentReference Page="3"/>
17 </Story>
18 </DocumentStructure>
```

19 *end example]*

20 *Example 16-4. Story flowing back and forth across a page boundary*

21 The following markup describes a page containing two tables, arranged side-by-side, each of  
22 which continues to the following page. In this case, the fragment is split and a fragment name  
23 is specified. `FragmentA` refers to the content leading up to the middle of the first (left) table and  
24 `FragmentB` is the continuation of this table on the following page. The flow then returns to the  
25 second (right) table on page 1 (`FragmentC`) before continuing with the rest of the story in  
26 `FragmentD`.

```
27 <DocumentStructure
28
29 xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
30 mentstructure">
31 <Story StoryName="Story_1">
32 <StoryFragmentReference FragmentName="FragmentA" Page="1"/>
33 <StoryFragmentReference FragmentName="FragmentB" Page="2"/>
34 <StoryFragmentReference FragmentName="FragmentC" Page="1"/>
35 <StoryFragmentReference FragmentName="FragmentD" Page="2"/>
36 </Story>
37 </DocumentStructure>
```

38 *end example]*

### 39 **16.1.2 StoryFragments Part**

40 The StoryFragments part contains content structure markup (describing such things as tables  
41 and paragraphs) for each story fragment that appears on the page. The content structure is  
42 expressed by tags that ultimately wrap `<NamedElement>` references that point to fixed page  
43 markup.

1 Table 16-1. StoryFragments part elements

Name	Description
<StoryFragments>	Root element.
<StoryFragment>	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.

2 Because a single content structural element can be split across pages, the <StoryBreak>  
3 element is provided to identify that a given element continues *to* the next story fragment or  
4 continues *from* a previous story fragment. A <StoryBreak> element MUST NOT be included in a  
5 position other than the first or last child element of a <StoryFragment> element [M9.12].

6 If a <StoryBreak> element is not present at the beginning of the content structure markup,  
7 consumers SHOULD consider the markup a continuation of the previous story fragment that  
8 must be merged [S9.4]. Likewise, if a <StoryBreak> element is not present at the end of the  
9 content structure markup, consumers SHOULD consider the markup a continuation to the next  
10 story fragment that must be merged to determine the cross-fragment content structure [S9.4].

11 Content structure is merged on an element-by-element basis, merging the last element closed  
12 in the leading story fragment with the first element opened in the trailing story fragment. This  
13 process continues until the closing tag from the leading story fragment no longer matches the  
14 opening tag from the trailing story fragment.

15 <TableCellStructure> elements require special merging, such that all <TableCellStructure>  
16 elements within a <TableRowStructure> element are merged. In order to merge the table cells  
17 and rows correctly, producers MUST specify empty <TableCellStructure> elements for cells that  
18 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
8     xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
9     mentstructure">
10     <StoryFragment FragmentType="Header">
11         <StoryBreak />
12         <ParagraphStructure>
13             <NamedElement NameReference="Block1" />
14         </ParagraphStructure>
15         <StoryBreak />
16     </StoryFragment>
17     <StoryFragment StoryName="Story1" FragmentType="Content">
18         <StoryBreak />
19         <SectionStructure>
20             <TableStructure>
21                 <TableRowGroupStructure>
22                     <TableRowStructure>
23                         <TableCellStructure>
24                             <ParagraphStructure>
25                                 <NamedElement NameReference="Block2" />
26                                 <NamedElement NameReference="Block3" />
27                             </ParagraphStructure>
28                         </TableCellStructure>
29                         <TableCellStructure>
30                             <ParagraphStructure>
31                                 <NamedElement NameReference="Block4" />
32                             </ParagraphStructure>
33                         </TableCellStructure>
34                     </TableRowStructure>
35                     <TableRowStructure>
36                         <TableCellStructure>
37                             <ParagraphStructure>
38                                 <NamedElement NameReference="Block5" />
39                                 <NamedElement NameReference="Block6" />
40                             </ParagraphStructure>
41                         </TableCellStructure>
42                         <TableCellStructure>
43                             <ParagraphStructure>
44                                 <NamedElement NameReference="Block7" />
45                             </ParagraphStructure>
46                         </TableCellStructure>
47                     </TableRowStructure>
48                 </TableRowGroupStructure>
49             </TableStructure>
50         </SectionStructure>
51     </StoryFragment>
52     <StoryFragment FragmentType="Footer">
53         <StoryBreak />

```

```

1      <ParagraphStructure>
2          <NamedElement NameReference="Block8" />
3      </ParagraphStructure>
4      <StoryBreak />
5  </StoryFragment>
6  </StoryFragments>

7
8  <!-- Second StoryFragments part -->
9
10 <StoryFragments
11
12 | xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
13 | mentstructure">
14     <StoryFragment FragmentType="Header">
15         <StoryBreak />
16         <ParagraphStructure>
17             <NamedElement NameReference="Block9" />
18         </ParagraphStructure>
19         <StoryBreak />
20     </StoryFragment>
21     <StoryFragment StoryName="Story1" FragmentType="Content">
22         <SectionStructure>
23             <TableStructure>
24                 <TableRowGroupStructure>
25                     <TableRowStructure>
26                         <TableCellStructure />
27                         <TableCellStructure>
28                             <ParagraphStructure>
29                                 <NamedElement NameReference="Block10" />
30                                 <NamedElement NameReference="Block11" />
31                             </ParagraphStructure>
32                         </TableCellStructure>
33                     </TableRowStructure>
34                     <TableRowStructure>
35                         <TableCellStructure>
36                             <ParagraphStructure>
37                                 <NamedElement NameReference="Block12" />
38                             </ParagraphStructure>
39                         </TableCellStructure>
40                         <TableCellStructure>
41                             <ParagraphStructure>
42                                 <NamedElement NameReference="Block13" />
43                             </ParagraphStructure>
44                         </TableCellStructure>
45                     </TableRowStructure>
46                 </TableRowGroupStructure>
47             </TableStructure>
48         </SectionStructure>
49         <StoryBreak />
50     </StoryFragment>
51     <StoryFragment FragmentType="Footer">
52         <StoryBreak />
53         <ParagraphStructure>
54             <NamedElement NameReference="Block14" />

```

```

1      </ParagraphStructure>
2      <StoryBreak />
3  </StoryFragment>
4  </StoryFragments>

5

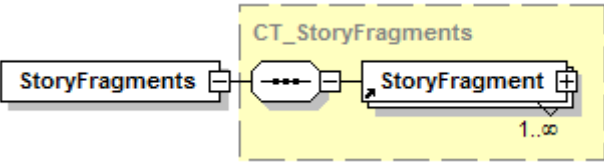
6  <!-- Resulting merged content structure for Story1 -->
7
8  <SectionStructure>
9    <TableStructure>
10     <TableRowGroupStructure>
11       <TableRowStructure>
12         <TableCellStructure>
13           <ParagraphStructure>
14             <NamedElement NameReference="Block2" />
15             <NamedElement NameReference="Block3" />
16           </ParagraphStructure>
17         </TableCellStructure>
18         <TableCellStructure>
19           <ParagraphStructure>
20             <NamedElement NameReference="Block4" />
21           </ParagraphStructure>
22         </TableCellStructure>
23       </TableRowStructure>
24       <TableRowStructure>
25         <TableCellStructure>
26           <ParagraphStructure>
27             <NamedElement NameReference="Block5" />
28             <NamedElement NameReference="Block6" />
29           </ParagraphStructure>
30         </TableCellStructure>
31         <TableCellStructure>
32           <ParagraphStructure>
33             <NamedElement NameReference="Block7" />
34             <NamedElement NameReference="Block10" />
35             <NamedElement NameReference="Block11" />
36           </ParagraphStructure>
37         </TableCellStructure>
38       </TableRowStructure>
39       <TableRowStructure>
40         <TableCellStructure>
41           <ParagraphStructure>
42             <NamedElement NameReference="Block12" />
43           </ParagraphStructure>
44         </TableCellStructure>
45         <TableCellStructure>
46           <ParagraphStructure>
47             <NamedElement NameReference="Block13" />
48           </ParagraphStructure>
49         </TableCellStructure>
50       </TableRowStructure>
51     </TableRowGroupStructure>
52   </TableStructure>
53 </SectionStructure>

```

1 *end example]*

2 **16.1.2.1 <StoryFragments> Element**

3 element **StoryFragments**

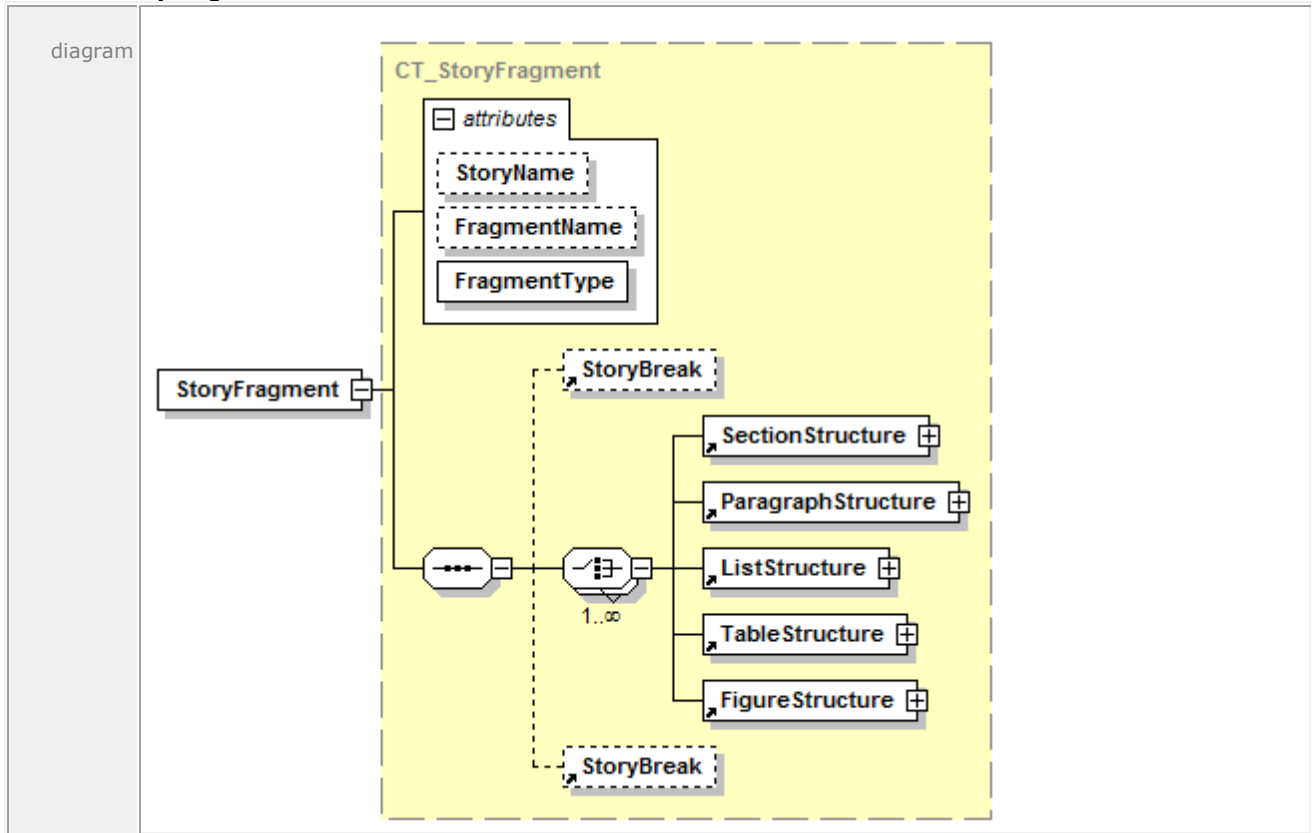
<p>diagram</p>	
<p>annotation</p>	<p>The root of a StoryFragments part. Contains all story fragments that appear on a specific page.</p>

4 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
11 |   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
12 |   mentstructure">
13     <StoryFragment FragmentType="Header">
14       <StoryBreak />
15       <ParagraphStructure>
16         <NamedElement NameReference="Block13" />
17         <NamedElement NameReference="Block14" />
18       </ParagraphStructure>
19       <StoryBreak />
20     </StoryFragment>
21     <StoryFragment StoryName="Story1" FragmentType="Content">
22       <StoryBreak />
23       <ParagraphStructure>
24         <NamedElement NameReference="Block1" />
25         <NamedElement NameReference="Block2" />
26       </ParagraphStructure>
27       <TableStructure>
28         <TableRowGroupStructure>
29           <TableRowStructure>
30             <TableCellStructure>
31               <ParagraphStructure>
32                 <NamedElement NameReference="Block3" />
33                 <NamedElement NameReference="Block4" />
34               </ParagraphStructure>
35             </TableCellStructure>
36             <TableCellStructure>
37               <ParagraphStructure>
38                 <NamedElement NameReference="Block5" />
39               </ParagraphStructure>
40             </TableCellStructure>
41           </TableRowStructure>
42         </TableRowGroupStructure>
43       </TableStructure>
44       <SectionStructure>
45         <ParagraphStructure>
46           <NamedElement NameReference="Block6" />
47         </ParagraphStructure>
48         <ParagraphStructure>
49           <NamedElement NameReference="Block7" />
50           <NamedElement NameReference="Block8" />
51         </ParagraphStructure>
52       </SectionStructure>

```

```

1      <SectionStructure>
2          <FigureStructure>
3              <NamedElement NameReference="Block9" />
4          </FigureStructure>
5          <ListStructure>
6              <ListItemStructure>
7                  <ParagraphStructure>
8                      <NamedElement NameReference="Block10" />
9                  </ParagraphStructure>
10             </ListItemStructure>
11             <ListItemStructure>
12                 <ParagraphStructure>
13                     <NamedElement NameReference="Block11" />
14                 </ParagraphStructure>
15             </ListItemStructure>
16             <ListItemStructure>
17                 <ParagraphStructure>
18                     <NamedElement NameReference="Block12" />
19                 </ParagraphStructure>
20             </ListItemStructure>
21         </ListStructure>
22     </SectionStructure>
23     <StoryBreak />
24 </StoryFragment>
25 <StoryFragment FragmentType="Footer">
26     <StoryBreak />
27     <ParagraphStructure>
28         <NamedElement NameReference="Block15" />
29         <NamedElement NameReference="Block16" />
30         <NamedElement NameReference="Block17" />
31     </ParagraphStructure>
32     <StoryBreak />
33 </StoryFragment>
34 </StoryFragments>

```

35 *end example]*

36 A <StoryFragment> element MAY be identified with a FragmentName attribute to distinguish it  
37 from other fragments for the same story on a single page [M2.72].

38 *Example 16–7. Story fragments markup using a fragment name*


```

39     <StoryFragments
40
41     xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/docu
42     mentstructure">
43         <StoryFragment
44             StoryName="Story1"
45             FragmentName="Fr1"
46             FragmentType="Content">
47             <StoryBreak />
48             <ParagraphStructure>
49                 <NamedElement NameReference="Block1" />
50                 <NamedElement NameReference="Block2" />
51             </ParagraphStructure>
52             <StoryBreak />

```

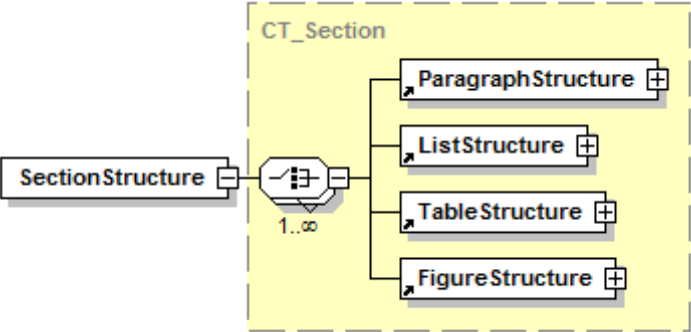
```
1      </StoryFragment>
2      <StoryFragment
3          StoryName="Story1"
4          FragmentName="Fr2"
5          FragmentType="Content">
6          <StoryBreak />
7          <ParagraphStructure>
8              <NamedElement NameReference="Block8" />
9          </ParagraphStructure>
10         <StoryBreak />
11     </StoryFragment>
12 </StoryFragments>
13 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>.

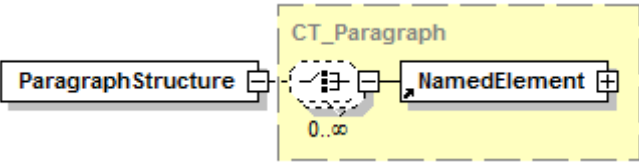
3 The <StoryBreak> element signals to the consumer not to perform merging across story  
4 fragments to determine the content structure.

5 **16.1.2.4 <SectionStructure> Element**6 element **SectionStructure**

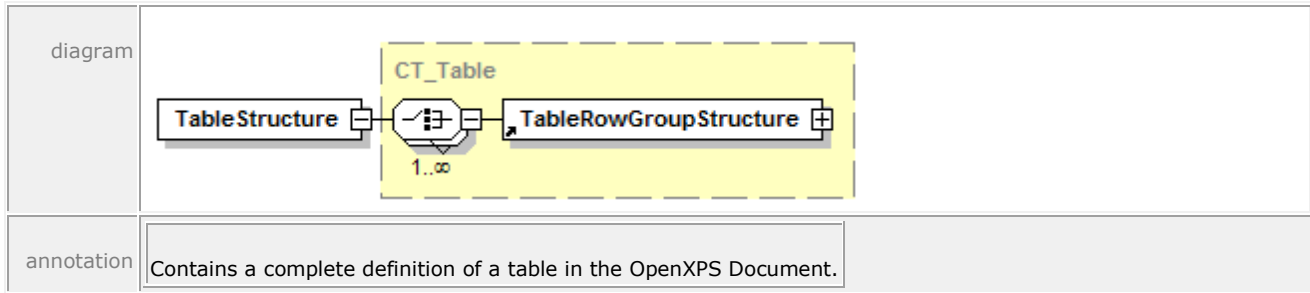
diagram	
annotation	Provides an arbitrary grouping of content structural markup elements.

7 The <SectionStructure> element provides an arbitrary grouping of <Paragraph>,  
8 <TableStructure>, <ListStructure>, and <FigureStructure> elements.

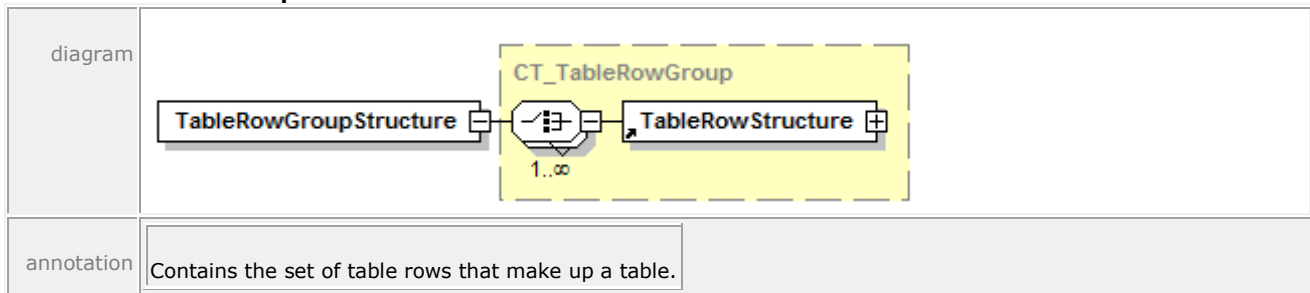
9 **16.1.2.5 <ParagraphStructure> Element**10 element **ParagraphStructure**

diagram	
annotation	Contains the named elements that constitute a single paragraph.

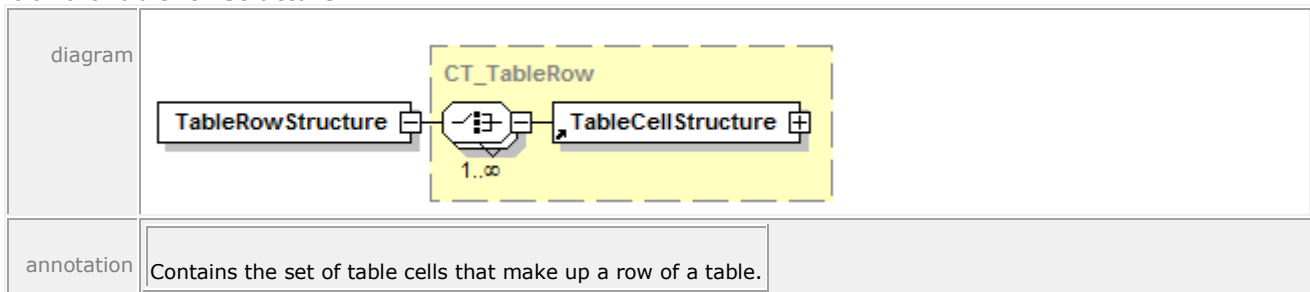
11 A <ParagraphStructure> element describes the list of <NamedElement> elements that  
12 constitute a single paragraph.

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**

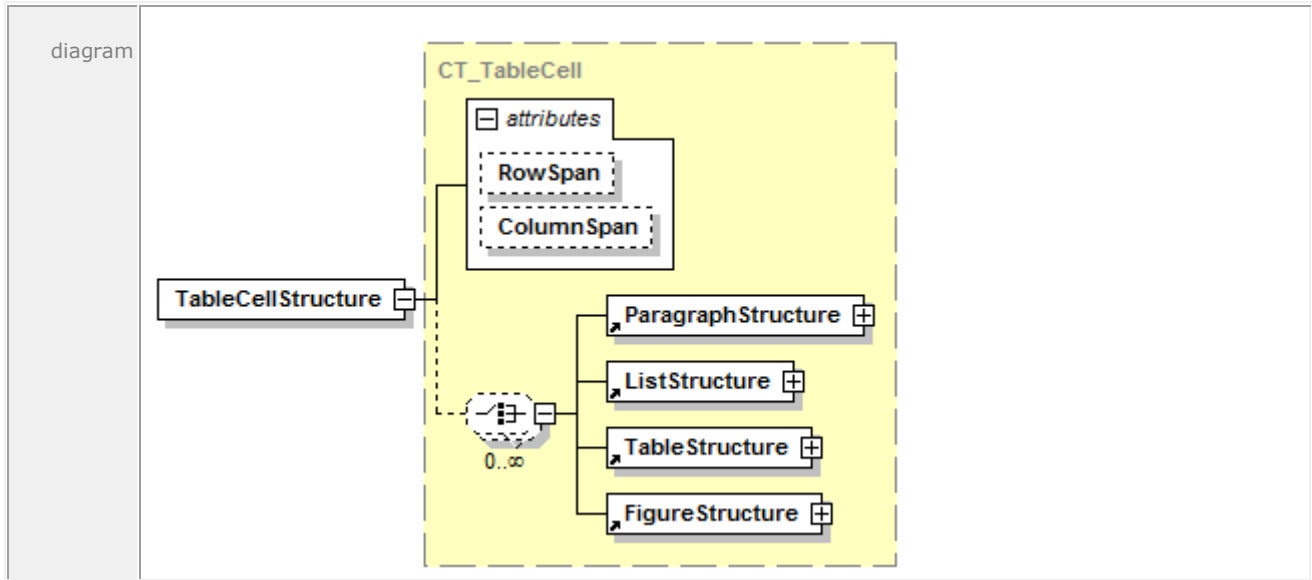
7 A <TableRowGroupStructure> element is REQUIRED in order to specify a set of  
 8 <TableRowStructure> elements [M9.13].

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**



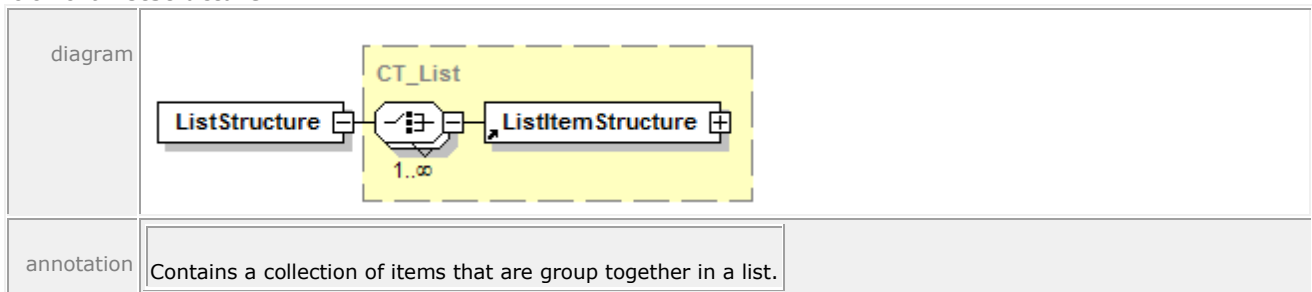
attributes	Name	Type	Use	Default	Fixed	Annotation
	RowSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of rows this cell spans, or merges into a single cell.
	ColumnSpan	<u>ST_TableSpan</u>	optional	1		Indicates the number of columns this cell spans, or merges into a single cell.

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 [09.16].

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**

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.					

3 A <ListItemStructure> element defines a single item in a list.

4 **16.1.2.12 <FigureStructure> Element**

5 element **FigureStructure**

diagram						
annotation	Groups the named elements that constitute a single drawing or diagram.					

6 A <FigureStructure> element includes a group of named elements that comprise a single  
 7 drawing or diagram.



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 OpenXPS 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 OpenXPS Document by specifying a value for the Name  
3 attribute. The name SHOULD be unique within the scope of the fixed document [S9.8]. If it is  
4 not unique, only 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 OpenXPS Documents specify two forms of URI fragment identifiers to address locations within  
24 an OpenXPS Document. The first is a named address. [*Example:*  
25 "http://xps/MyPackage#MyAddress", where "http://xps/MyPackage" is an OpenXPS Document  
26 and "MyAddress" is a named address within the document. *end example]* The second is an  
27 absolute page number within the OpenXPS Document. [*Example:* "http://xps/MyPackage#15",  
28 where "15" references the FixedPage part associated with the fifteenth <PageContent> entry  
29 among all the fixed documents in the fixed document 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 OpenXPS Document has a 3-page fixed document,  
32 followed by a 10-page fixed document, followed by an 8-page fixed document, the fragment  
33 identifier "#15" refers to the second page of the third fixed document in the fixed document  
34 sequence. *end example]* Internal references MUST specify a page address relative to the fixed  
35 document 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 OpenXPS Documents SHOULD support hyperlinks [S9.6].

7 The Name value, if specified, MUST meet the following requirements [M9.14]:

- 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 [M9.14].
- 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 [M9.14].

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 OpenXPS Document package, respectively. The base URI used to  
20 resolve a relative URI is that of the FixedPage part in which the element with the  
21 FixedPage.NavigateUri attribute appears. Therefore, a hyperlink to a destination within the fixed  
22 document of the source MUST specify the destination in the context of the FixedDocument part  
23 [M9.4]. [*Example*: “../FixedDoc1.fdoc#MyDestination”. *end example*] A destination in the same  
24 fixed document SHOULD be expressed as a relative URI [S9.11].

25 The FixedPage.NavigateUri attribute is OPTIONAL [O9.17]. 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 OpenXPS Documents SHOULD support  
28 hyperlinks [S9.6].

---

## 16.3 Selection

Viewing consumers that support interactivity MAY support selection and copying [O9.9].

Selection order within an OpenXPS 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 OpenXPS 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 OpenXPS 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].
- Within a story, order `<StoryFragmentReference>` elements in the sequence they appear in the `DocumentStructure` part. Producers SHOULD order `<StoryFragmentReference>` elements in their intended reading order [S9.21].

- 1       • Within a story fragment, order by <NamedElement> references in the StoryFragments  
2       part markup.
- 3       • Append all un-referenced elements that appear in the fixed page markup, ordered by  
4       page number, then markup order.

### 5       **16.4.2 Screen Reader Applications**

6       Screen reader applications read the contents of the document aloud. A screen reader consumer  
7       SHOULD read the document according to its reading order [S9.22]. The application SHOULD  
8       use the UnicodeString attribute of each <Glyphs> element [S9.23]. In addition, screen readers  
9       MAY inspect the Indices attribute to resolve potential ambiguities [O9.13].

10      If the screen reader provides features to navigate the document by structural elements, such as  
11      paragraphs or table rows, it SHOULD use any document structure information included in the  
12      OpenXPS Document [S9.24].

13      If the screen reader provides features to describe images, it SHOULD read the text provided in  
14      the AutomationProperties.Name and AutomationProperties.HelpText attributes [S9.25].

15      If the screen reader provides features to describe hyperlink addresses, it SHOULD read the text  
16      provided in the FixedPage.NavigateUri attribute [S9.26].

### 17      **16.4.3 Text Alternatives for Graphics and Images**

18      Images and graphics SHOULD specify text alternatives for images and graphics to make this  
19      content accessible to vision-impaired individuals [S9.27]. There are short and long textual  
20      descriptions, specified in the AutomationProperties.Name and AutomationProperties.HelpText  
21      attributes of <Path> and <Canvas>, respectively.

22      The AutomationProperties.Name attribute SHOULD contain a short description of the basic  
23      contents of the image or vector graphic [S9.27]. [*Example: "A sitting dog." end example*]The  
24      AutomationProperties.HelpText attribute can contain a more detailed description of the image or  
25      graphic. [*Example: "A cocker spaniel with brown eyes, golden fur, and its tongue hanging out.  
26      It is sitting on a beanbag directly facing the camera." end example*]

27      An image SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText  
28      attributes on the <Path> element that is filled with an <ImageBrush> [S9.28]. These attributes  
29      describe the content specified by the ImageSource attribute of the <ImageBrush> element.

30      A vector graphic (a collection of one or more <Path> elements representing a single drawing)  
31      SHOULD specify the AutomationProperties.Name and AutomationProperties.HelpText attributes only  
32      once, directly on a <Canvas> element wrapping the <Path> elements comprising the graphic  
33      [S9.29].

34      Individual <Path> elements that do not provide any semantic meaning (such as a line between  
35      sections or outlining a table) SHOULD NOT specify these text alternative attributes [S9.27].

36



## 17. OpenXPS Document Package Features

The OpenXPS Document format extends package-level interleaving and digital signatures as described in the OPC Standard.

---

### 17.1 Interleaving Optimizations

Interleaving concerns the physical organization of OpenXPS Documents, rather than their logical structure. It allows consumers to process linearly 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 OpenXPS Document is interleaved, these rules SHOULD be followed:

- The Content Types stream SHOULD be interleaved according to the recommendations in the OPC Standard [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].
- The relationships for the DiscardControl part and the StartPart SHOULD both be written in the first piece of the package relationship part, and that piece SHOULD be before the first FixedPage part in the package [S10.20].
- The piece of the DiscardControl part that includes a Discard element with a SentinelPage attribute referencing a FixedPage part SHOULD be written to the package before that FixedPage part [S10.21].

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 process correctly 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].

- 1 Consumers can choose to parse an OpenXPS Document in a head-first or tail-first manner. Tail-  
2 first parsing reveals certain package errors earlier, such as inconsistencies between the ZIP  
3 central directory and local file headers. Head-first OpenXPS Document consumers SHOULD  
4 attempt to detect inconsistent packages as soon as possible and SHOULD instantiate an error  
5 condition, even if they have already processed the pages that resulted in the error [S10.18].  
6 Head-first consumers that discard parts would need to retain the name and length of any  
7 discarded part to comply with this recommendation.
- 8 [*Note*: Streaming and handling of discard control are complicated significantly by any  
9 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]. The content of an empty PrintTicket is implementation-defined (see  
8 §9.1.9).

### 9 **17.1.2 Optimizing Interleaving Order**

10 Producers MAY optimize the interleaving order of parts to help consumers avoid stalls during  
11 read-time streaming, and to allow consumers to manage their memory resources more  
12 efficiently [O10.2].

13 The optimization strategy is suggested by the consumer architecture. Therefore, interleaving  
14 optimization is typically implemented by a software component such as a driver or filter that is  
15 specific to (or aware of) the consumer architecture.

#### 16 **17.1.2.1 Single-Threaded Parsing Architectures**

17 An optimal interleaving scheme for consumers with a single-threaded parsing model interleaves  
18 parts so that each part that is required to consume a single page (FixedPage, images, and  
19 fonts) is contained in the package in its entirety, prior to the FixedPage part being referenced  
20 from the FixedDocument part's markup.

21 Single-threaded parsing architectures typically require more run-time memory resources than  
22 multi-threaded parsing architectures because the context in which a resource is used is  
23 unknown at the time the resource is received. This requires deferred processing and additional  
24 buffering.

25 [*Note:* When interleaving entities containing XML markup, such as the DiscardControl part, the  
26 Content Types stream, and the FixedDocument part, there is no guarantee that XML element  
27 boundaries will align with piece boundaries in the physical package. This adds a complexity to  
28 single-threaded parsing architectures: the parser must be pre-emptable. Certain existing XML  
29 parser implementations might require a pre-tokenization step. *end note*]

30 *Example 17-1. Optimized interleaving for a single-threaded parsing architecture*

31 The following markup describes a sequence of two fixed documents, the first having two  
32 FixedPage parts and the second having one FixedPage part:

33

Part/Piece	Markup
Font1.ttf	...binary font data...
Other resources	...resource data...
Page1	<FixedPage xmlns="http://schemas.openxps.orgschemas.microsoft.c om /oxps/v1.0/xps/2005/06" ...> <Glyphs FontURI="Font1.ttf"/>

	</FixedPage>
Page1.rels	<Relationships xmlns= "http://schemas.openxmlformats.org/package/2006/relationships">  <Relationship Type= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0 /required-resource" Target="Font1.ttf"/>  </Relationships>
FixedDocument1/[0].piece	<FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0">  <PageContent Source="Page1"/>
Sequence1/[0].piece	<FixedDocumentSequence xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0">  <DocumentReference Source="FixedDocument1"/>
_rels/.rels/[0].piece	<Relationships xmlns= "http://schemas.openxmlformats.org/package/2006/relationships">  <Relationship Type="StartPart" Target="Sequence1"/>
Page2	<FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0" ...>...</FixedPage>
FixedDocument1/[1].last.piece	<PageContent Source="Page2"/> </FixedDocument>
Page3	<FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0" ...>...</FixedPage>
FixedDocument2	<FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0">  <PageContent Source="Page3"/>  </FixedDocument>
Sequence1/[1].last.piece	<DocumentReference Source="FixedDocument2" /> </FixedDocumentSequence>
_rels/.rels/[1].last.piece	</Relationships>

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.

- 1 [Note: When interleaving entities containing XML markup, such as the DiscardControl part, the
- 2 content type stream, and the FixedDocument part, there is no guarantee that XML element
- 3 boundaries will align with piece boundaries in the physical package. A multi-threaded parsing
- 4 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>&lt;Relationships xmlns="http://schemas.openxmlformats.org/schemas/microsoft.com/package/2006/relationships"&gt; &lt;Relationship Type="StartPart" Target="Sequence1"/&gt;</pre>
Sequence1/[0].piece	<pre>&lt;FixedDocumentSequence xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxmlformats.org/oxps/v1.0"&gt; &lt;DocumentReference Source="FixedDocument1"/&gt;</pre>
FixedDocument1/[0].piece	<pre>&lt;FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxmlformats.org/oxps/v1.0"&gt; &lt;PageContent Source="Page1"/&gt;</pre>
Page1.rels	<pre>&lt;Relationships xmlns= "http://schemas.openxmlformats.org/package/2006/relationships"&gt; &lt;Relationship Type= "http://schemas.microsoft.com/xps/2005/06/schemas.openxmlformats.org/oxps/v1.0/required-resource" Target="Font1.ttf"/&gt; &lt;/Relationships&gt;</pre>
Page1	<pre>&lt;FixedPage ="http://schemas.openxmlformats.org/schemas/microsoft.com/oxps/v1.0/xps/2005/06" ...&gt; &lt;Glyphs FontURI="Font1.ttf"/&gt; &lt;/FixedPage&gt;</pre>
Font1.ttf	...binary font data...
Other resources	...resource data...
FixedDocument1/[1].last.piece	<PageContent Source="Page2"/>
Page2	<pre>&lt;FixedPage xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxmlformats.org/oxps/v1.0" ...&gt;...&lt;/FixedPage&gt;</pre>
FixedDocument1/[2].last.piece	</FixedDocument>
Sequence1/[1].last.piece	<pre>&lt;DocumentReference Source="FixedDocument2" /&gt; &lt;/FixedDocumentSequence&gt;</pre>
FixedDocument2	<pre>&lt;FixedDocument xmlns= "http://schemas.microsoft.com/xps/2005/06/schemas.openxmlformats.org/oxps/v1.0"&gt;</pre>

---

```

nxps.org/oxps/v1.0">
<PageContent Source="Page3"/>
</FixedDocument>
Page3
<FixedPage
xmlns="http://schemas.microsoft.com/xps/2005/06/schem
as.openxps.org/oxps/v1.0" ...>...</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].

5 To address resource constraints:

- 6 • Consumers MAY discard FixedPage parts once they have been processed [O10.3]
- 7 • Consumers MAY discard FixedDocument and FixedDocumentSequence parts after all their  
8 child elements and their closing tags have been processed [O10.4].
- 9 • In the absence of explicit directives to the contrary (see §17.1.4), consumers MAY  
10 discard parts as directed by the DiscardControl part [O10.5]. Consumers MUST NOT  
11 discard any other parts [*Example*: Such as parts containing fonts, images, or other  
12 resources *end example*] unless they have the ability to access the parts again [M10.4].

13 If a consumer encounters a reference to an unknown part, it MUST continue to receive further  
14 bytes of the package until the unknown part has been transmitted *or* until the end of the  
15 package is reached (indicating an error condition) [M10.5]; if the end of the package is reached  
16 the consumer SHOULD instantiate an error condition [S10.23].

### 17 **17.1.4 Consumers with Resource Constraints**

18 To produce an OpenXPS Document for streaming consumption by consumers with limited  
19 memory resources, some producers MAY choose a suitable interleaving order by modeling the  
20 resource management behavior of the consumer [O10.6]. These producers, referred to as  
21 *drivers*, must have specific knowledge of the OpenXPS Document consumer. Due to resource  
22 constraints, some consumers are unable to consume arbitrary OpenXPS Documents and always  
23 require assistance from an external driver.

24 When some consumers with limited memory resources receive a OpenXPS Document in a  
25 streaming fashion, there might be an opportunity to discard parts when necessary and reload  
26 them again when needed. Producers, such as drivers, that target such consumers SHOULD  
27 follow these steps [S10.7]:

- 28 • Conservatively model the memory usage of the device.
- 29 • Interleave pieces of parts in the correct order.
- 30 • Decide when certain parts can be discarded by the consumer and inform the consumer  
31 within the package stream (see §17.1.4.1).
- 32 • Add to the package a uniquely named copy of a resource that could have been discarded,  
33 if the resource is referenced by a part sent later in the stream. Those later references  
34 are also updated to refer to the new copy of the resource.

1 **17.1.4.1 DiscardControl Part**

2 In addition to optimally ordering interleaved parts, producers can support consumers with  
 3 resource constraints by means of the DiscardControl part. The DiscardControl part is a well-  
 4 known part containing a list of resources that are safe for the consumer to discard.  
 5 DiscardControl parts are stored in OpenXPS Documents in an interleaved fashion, allowing a  
 6 resource-constrained consumer to discard a part when that part is no longer required to  
 7 process pages in the payload, as soon as it appears in the DiscardControl part. DiscardControl  
 8 parts are targeted with a DiscardControl package relationship, as specified in §A. There MUST  
 9 NOT be more than one DiscardControl package relationship [M10.23]. The DiscardControl part  
 10 MUST NOT reference itself [M10.6]; doing so is considered an error.

11 ~~DiscardControl parts that are not well-formed SHOULD NOT be processed and an error~~  
 12 ~~condition SHOULD NOT be instantiated [S10.8]. Consumers MAY elect not to instantiate an~~  
 13 ~~error condition when encountering DiscardControl parts that do not conform to this specification~~  
 14 ~~[Ox.xx].~~ The consumer MAY decide to ignore the malformed DiscardControl part in its entirety  
 15 or from the first malformed node onward [O10.7].

16 In some cases, producers might rewrite the contents of a package so that parts are provided  
 17 more than once, allowing consumers to discard a part in order to free resources for additional  
 18 processing. Each instance of a part MUST be stored as a new, uniquely named part in the  
 19 package [M10.24].

20 *Example 17-3. A DiscardControl part*

```
21 <DiscardControl
22   xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/disc
23   ard-
24   control">
25   <!-- May discard partname1 as soon as starting to process
26     page11.xaml -->
27   <Discard SentinelPage="/page11.xaml" Target="/partname1" />
28   <!-- May discard partname2 as soon as starting to process
29     page13.xaml -->
30   <Discard SentinelPage="/page13.xaml" Target="/partname2" />
31   ...
32 </DiscardControl>
```

33 *end example]*

34 **17.1.4.1.1 <DiscardControl> Element**

35 element **DiscardControl**

diagram	
annotation	Contains a list of resources that are safe for a consumer to discard.

36 **17.1.4.1.2 <Discard> Element**

37 element **Discard**

diagram	<p>The diagram illustrates the relationship between a &lt;Discard&gt; element and a &lt;CT_Discard&gt; element. The &lt;Discard&gt; element has two attributes: Target and SentinelPage. The &lt;CT_Discard&gt; element also has two attributes: SentinelPage and Target. The SentinelPage attribute of &lt;CT_Discard&gt; is linked to the SentinelPage attribute of &lt;Discard&gt;, and the Target attribute of &lt;CT_Discard&gt; is linked to the Target attribute of &lt;Discard&gt;. A dashed box labeled 'attributes' is shown next to the SentinelPage attribute of &lt;CT_Discard&gt;.</p>					
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 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 parts. 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].

---

## 17.2 Digital Signatures

The digital signature specification for OpenXPS Documents is described in the OPC Standard. It allows users to sign arbitrary parts, relationship parts, and individual relationships. Although OpenXPS Documents also use these digital signature mechanisms, they have a specific signature policy and a specific signing mechanism for documents containing co-signing requests.

[\[Note: Consistent with the OPC Specification, implementations may include signatures with arbitrary data in the XML Signature <Object> element. end note\]](#)

### 17.2.1 Signature Policy

This Standard defines the signature policy that governs the methods of signing and verifying signatures for OpenXPS Documents. The OpenXPS 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 OpenXPS Document MUST be considered signed according to the OpenXPS 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 OpenXPS 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.



- 1 h. All SignatureDefinitions parts associated via a Signature Definitions relationship with  
2 any signed FixedDocument part. (Once a document is signed, adding any new  
3 signature definitions will invalidate the signature.)
  - 4 i. All Thumbnail parts associated via a Thumbnail relationship from the package root or  
5 with any signed FixedPage ~~or FixedDocument~~ part.
  - 6 2. The following parts MAY be signed [O10.16]:
    - 7 a. The CoreProperties part.
    - 8 b. The Digital Signature Origin part.
    - 9 c. A Digital Signature Certificate part.
    - 10 d. PrintTicket parts.
    - 11 e. DiscardControl parts.
  - 12 3. All relationships with the following RelationshipTypes (see §A) MUST be signed [M10.10]:
    - 13 a. StartPart relationship from the package root
    - 14 b. DocumentStructure relationship from a FixedDocument part
    - 15 c. StoryFragments relationship from a FixedPage part
    - 16 d. Digital Signature Definitions relationship from a FixedDocument part
    - 17 e. Required Resource relationship from a FixedPage part
    - 18 f. Restricted Font relationship from a FixedDocument part
    - 19 g. Thumbnail relationship from a FixedPage part, ~~a FixedDocument part,~~ or the package  
20 root
  - 21 4. All relationships with the following RelationshipTypes MUST be signed if their Target part  
22 is signed [M10.10]:
    - 23 a. Core Properties relationship
    - 24 b. Digital Signature Origin relationship
    - 25 c. Digital Signature Certificate relationship from a Digital Signature XML Signature part
    - 26 d. PrintTicket relationship
    - 27 e. DiscardControl relationship
  - 28 5. Relationships with the following RelationshipTypes MAY be signed as a group (they MUST  
29 NOT be signed individually) [M10.10]:
    - 30 a. All Digital Signature XML Signature relationships from the Digital Signature Origin part  
31 (signing all relationships of this RelationshipType will cause this signature to break  
32 when a new signature is added).
  - 33 6. All of the above-referenced parts and relationships MUST be signed using a single digital  
34 signature [M10.10].
- 35 An OpenXPS Document MUST NOT be considered signed according to the OpenXPS Document  
36 signing policy if [M10.11]:
- 37 1. Any part not covered by the signing rules above is included in the signature.
  - 38 2. Any relationship not covered by the signing rules above is included in the signature.

1 An OpenXPS Document digital signer MUST NOT sign an OpenXPS Document that contains  
2 content (parts or relationships parts) to be signed that defines the Markup Compatibility  
3 namespace when the signer does not fully understand all elements, attributes, and alternate  
4 content representations introduced through the markup compatibility mechanisms [M10.12]. An  
5 OpenXPS Document digital signer MAY choose not to sign any content (parts or relationships  
6 parts) that defines the Markup Compatibility namespace, even when the content is fully  
7 understood [O10.8].

8 An OpenXPS Document digital signer MUST NOT sign a PrintTicket part if it does not fully  
9 understand the PrintTicket content [M10.25].

#### 10 **17.2.1.2 Signing Validity**

11 An OpenXPS Document digital signature MUST be ~~treated shown~~ as an *incompliant digital*  
12 *signature* if [M10.13]:

- 13 • It violates any of the signing rules described above regarding parts or relationships that  
14 ~~MUST or~~ MUST NOT be signed.

15 An OpenXPS Document digital signature MUST be shown as a *broken digital signature* if  
16 [M10.14]:

- 17 • It is not an incompliant digital signature and it violates any of the signing rules described  
18 above regarding parts or relationships that MUST be signed.
- 19 • It is not an incompliant digital signature, but the signature fails the signature validation  
20 routines described in the OPC.

21 An OpenXPS Document digital signature MUST be shown as a *questionable digital signature* if  
22 any of the following are true [M10.15]:

- 23 • It is not an incompliant or broken digital signature, but the certificate cannot be  
24 authenticated against the certificate authority.
- 25 • It is not an incompliant or broken digital signature, but the signed content (parts and  
26 relationships) contain elements or attributes from an unknown namespace introduced  
27 through the Markup Compatibility mechanisms.

28 An OpenXPS Document digital signature MAY be shown as a questionable digital signature if  
29 [O10.9]:

- 30 • It is not an incompliant or broken digital signature, but contains some other detectable  
31 problem at the discretion of the consumer.

32 An OpenXPS Document digital signature MUST be shown as a *valid digital signature* if  
33 [M10.16]:

- 34 • It is not an incompliant, broken, or questionable digital signature.

#### 35 **17.2.1.3 Adding Signatures**

36 OpenXPS Documents MAY be signed more than once [O10.10]. A user who signs an OpenXPS  
37 Document might or might not want to allow any additional signing of the document. To prohibit  
38 additional signatures in an OpenXPS Document, the signing application MUST sign all the Digital  
39 Signature Origin part's relationships of relationship type Digital Signature with the same  
40 signature as the rest of the content [M10.17].

#### 1 **17.2.1.4 Certificate Store**

2 OpenXPS Document signatures MUST NOT refer to a remote certificate store (certificate not  
3 contained in the OpenXPS Document). All certificates MUST be stored in the OpenXPS  
4 Document either as a Certificate part or in the Digital Signature XML Signature part [M10.18].

#### 5 **17.2.1.5 Printing Signed Documents**

6 Consumers that support printing of signed documents SHOULD support control through  
7 PrintTicket settings pertaining to the treatment of OpenXPS Documents with invalid or  
8 questionable signatures [S10.22].

9 This setting can specify behaviors such as:

- 10 1. Print the job regardless of the validity of the digital signatures. Digital signatures can be  
11 ignored.
- 12 2. Print the job regardless of the validity of the digital signatures. In the event an invalid  
13 signature is encountered, an error page should print at the end of the job. Digital signatures  
14 cannot be ignored.
- 15 3. Print the job only if all digital signatures are valid. Digital signatures cannot be ignored.

#### 16 **17.2.2 Signature Definitions**

17 In some workflow scenarios, documents must be signed as a means of approving their content.  
18 [Example: Document producers might be required to sign their documents in order to provide  
19 proof of authenticity. end example] In other cases, reviewers might be required to co-sign  
20 content before it can be submitted for publication. These requirements can be fulfilled with a  
21 digitally signed OpenXPS Document.

22 Whereas the OpenXPS package model supports the signing of arbitrary content in a package,  
23 an OpenXPS Document signing workflow requires additional features, including the ability to  
24 specify co-signature requirements and to include workflow-specific signature information in the  
25 document. OpenXPS Document authors and signing parties provide such information in an XML  
26 *signature definition*.

27 Signature definitions are represented by <SignatureDefinition> elements within a single  
28 <SignatureDefinitions> element.

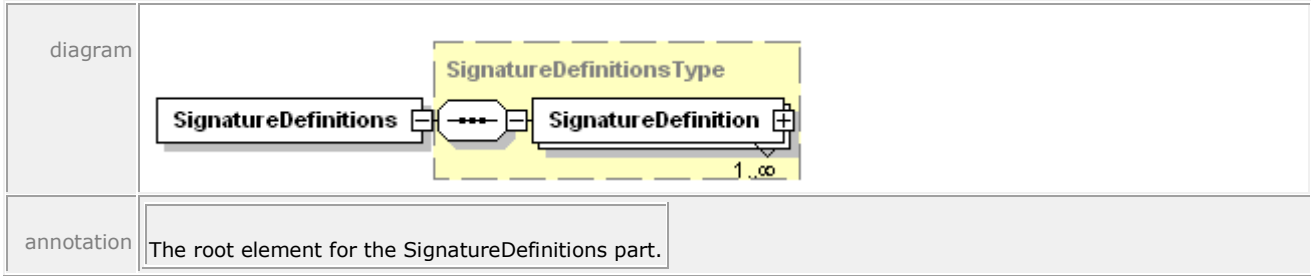
29 Example 17–4. A SignatureDefinitions part

```
30 <SignatureDefinitions
31 xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/
32 signature-definitions">
33 <SignatureDefinition SignerName="John SmithDorena-Paschke"
34 SpotID="0e0a7abb-48c9-595d-77db-305e84a05fc3">
35 <SpotLocation
36 PageURI="/Documents/1/Pages/2.fpage"
37 StartX="0.0"
38 StartY="0.0" />
39 <Intent>I have read and agree</Intent>
40 <SignBy>2005-08-20T23:59:59Z</SignBy>
41 <SigningLocation>New York, NYRedmond, WA</SigningLocation>
42 </SignatureDefinition>
43 </SignatureDefinitions>
```

1 *end example]*

2 **17.2.2.1 <SignatureDefinitions> Element**

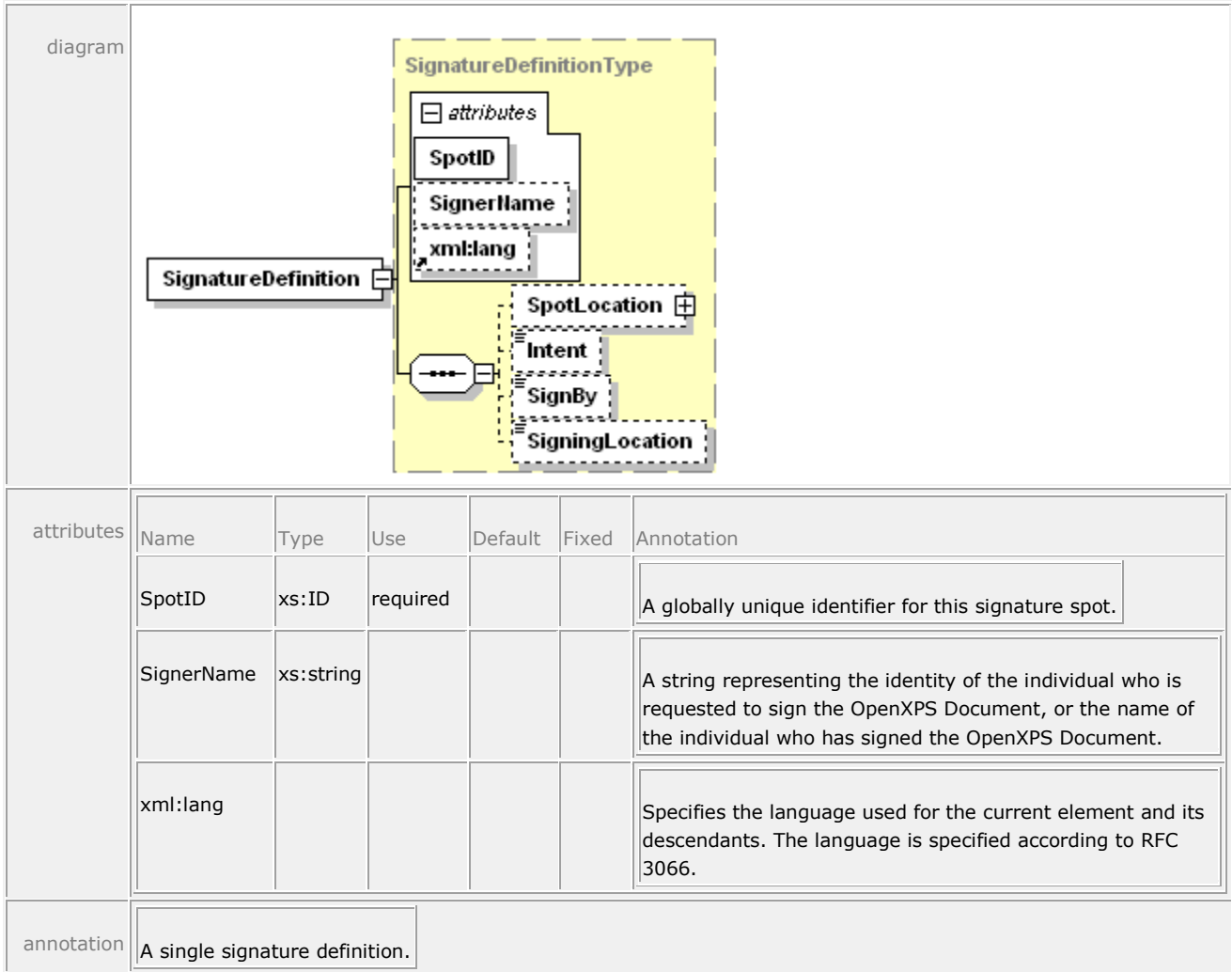
3 element **SignatureDefinitions**



4 If the SignatureDefinitions part exists, it MUST contain only one <SignatureDefinitions>  
 5 element [M10.26]. The XML namespace for the <SignatureDefinitions> element is specified  
 6 in §D.1.

7 **17.2.2.2 <SignatureDefinition> Element**

8 element **SignatureDefinitionsType/SignatureDefinition**



1 If the SignatureDefinitions part exists, there MUST be *at least* one <SignatureDefinition>  
2 element [M10.27].

### 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 [M10.29]. 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 Standard.~~

### 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.					

13 The <SpotLocation> element is OPTIONAL [O10.15]. It specifies where an OpenXPS Document  
14 viewer should place a visual representation or *signature spot* to indicate that a digital signature  
15 has been applied or requested. The viewing consumer SHOULD use the values specified in this  
16 element [O10.15]. Due to space and rendering limitations, producers MUST NOT assume that  
17 consumers will use these values [M10.20]. If the location specified by this element is not used,  
18 it is RECOMMENDED that consumers choose a location that does not contain any page content  
19 [S10.13].

20 The size and shape of the signature spot are determined by the consumer. Consumers MAY  
21 choose a size and shape based on the desired display information and page content [O10.13].  
22 However, it is RECOMMENDED that they render signature spots as consistently sized rectangles

1 that include the signer name, the intent, the signing location, and the scope of the OpenXPS  
 2 Document to be signed [S10.14]. It is also RECOMMENDED that the signature spot be a  
 3 clickable area used to launch the digital signing process [S10.15].


4 *Figure 17–1. A sample signature spot*



5

#### 6 **17.2.2.4 <Intent> Element**

7 element **SignatureDefinitionType/Intent**

diagram	
annotation	A string that represents the intent to which the signing party agrees when signing the document.

8 Consumers MUST display the full value of the <Intent> element to the signing party, either in  
 9 the signature spot or through some other mechanism [M10.21].

10 [Note: Consumers that wish to display signature spots must consider the implications of  
 11 supporting any Unicode character that can be specified in the <Intent> element, and of the  
 12 possibility of Unicode non-characters being included. They must also make decisions about the  
 13 appropriate font face and size to use as well as determine the proper layout and interactivity of  
 14 the signature spot. In the interests of maximizing compatibility, creators are recommended to  
 15 normalize the string using NFC. These decisions are implementation-defined. *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 OpenXPS 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 OpenXPS Document  
 11 or by the signing party at the time of [requesting a signature signing](#) [O10.14].

12 **17.3 Core Properties**

13 OpenXPS 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 OpenXPS Document markup across various implementations. Producers MUST generate OpenXPS 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 OpenXPS 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 page 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.

#### 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  $\text{coord}_r$  expresses a real-number coordinate value and  $\text{coord}_d$  expresses a device coordinate value.

### 1 **18.1.3 Transforms**

2 OpenXPS Document markup supports affine transforms as expressed through the  
3 RenderTransform and Transform properties. An affine transform is represented as a list of six  
4 real numbers: m11, 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 [M11.10].

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-defined:

- 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

OpenXPS Document markup does not assume fixed implementation limits. However, consumers can have specific implementation limits imposed by their operating environment. OpenXPS 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]. If a consumer encounters markup with characteristics outside its implementation-defined limits, it MUST instantiate 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 OpenXPS 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 <sup>12</sup>	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 <sup>-12</sup>	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
<b>&lt;Glyphs&gt; element</b>			
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 OpenXPS Document markup per page	Bytes	64,000,000	Total size of markup after removing all unnecessary whitespace (according to the schema in A.1), 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 consumers SHOULD blend colors  
26 between gradient stops in the color space indicated by the ColorInterpolationMode attribute of the  
27 gradient brush, unless a PrintTicket setting provides an alternative blending color space that the  
28 consumer understands (see §9.1.9 and §15.5) [S11.13]. If none of the gradient stop elements  
29 uses an sRGB or scRGB color specification and the consumer understands the blending color  
30 space PrintTicket setting, the blending color space PrintTicket setting SHOULD be used  
31 [S11.13].

32 The function used for blending is:

33  $\text{BLEND}(\text{offset}, c_{10}, c_{hi})$

34 Where the offset is between 0 and 1.  $c_{10}$  and  $c_{hi}$  designate the color values for an offset of 0  
35 and 1, respectively.

36 If a ColorInterpolationMode value of SRgbLinearInterpolation is used, the BLEND() function  
37 SHOULD convert the color values to sRGB first, and then perform a linear interpolation between  
38 them [S11.14].

39 If a ColorInterpolationMode value of ScRgbLinearInterpolation is used, the BLEND() function  
40 SHOULD convert the color values to scRGB first, and then perform a linear interpolation  
41 between them [S11.15].

42 In the presence of transformations or when individual gradient stops are very close (separated  
43 by a few pixels or less in the device space), the local color gradient at the offset used in the  
44 BLEND() function might be large, resulting in a large change over the extent of a single device

1 pixel. In this case, it is RECOMMENDED that the BLEND() function interpolate the gradient over  
 2 the extent of each device pixel [S11.16]. However, the behavior MAY differ from this  
 3 recommendation in an implementation-defined manner [O11.7] and, therefore, producers  
 4 SHOULD NOT rely on a specific effect for such dense gradient specifications [S11.16].

5 As a consequence of this interpolation, radial gradients that define the gradient origin on or  
 6 outside ellipse create an "outside" area that can be rendered inconsistently. The radial  
 7 gradients that are affected are those that define multiple gradient stops that are of different  
 8 colors and are very close in Offset value to 0.0 or 1.0 (the gradient end points), for radial  
 9 gradients with a SpreadMethod value of Repeat or Reflect, respectively. For these affected  
 10 gradients, consumers MAY use an interpolated color value for the outside area [O11.8].  
 11 Depending on the resolution, this can result in different colors than those defined by the  
 12 gradient end points. The closer a gradient stop is to the affected gradient end point, the more  
 13 the rendering results on different consumers and at different display resolutions can differ.  
 14 Producers SHOULD therefore either avoid such close gradient stops to the gradient end point  
 15 when specifying radial gradients where the outside area is visible or avoid specifying radial  
 16 gradients with a gradient origin on or outside the ellipse (in which case there is no outside area)  
 17 to ensure consistent rendering results [S11.17].

### 18 18.3.2 Linear Gradients

19 Consumers SHOULD render an element filled with a linear gradient brush such that the  
 20 appearance is the same as if the following steps had been taken [S11.33]:

- 21 1. Transform the StartPoint and EndPoint attribute values using the current effective render  
 22 transform (including the render transform for the element being filled by the linear  
 23 gradient brush and the brush's transform itself).
- 24 2. If the SpreadMethod value is Pad, the colors of points on the line defined by the StartPoint  
 25 and EndPoint attributes are defined by interpolating the coordinates linearly, and each  
 26 color component (such as R, G, B for sRGB and scRGB) as well as the alpha component  
 27 is interpolated between the component values of the closest enclosing gradient stops:

```
28 For each offset (real number)  $t < 0$ :
29 {
30    $x(t) = (EndPoint_x - StartPoint_x) * t + StartPoint_x$ 
31    $y(t) = (EndPoint_y - StartPoint_y) * t + StartPoint_y$ 
32    $c(t) = c_{first}$ 
33    $a(t) = a_{first}$ 
34 }
```

35 Where  $c$  is the color component and  $a$  is the alpha component.  $c_{first}$  are the color  
 36 component values of the first gradient stop (after sorting) and  $a_{first}$  is the alpha  
 37 component value at the first gradient stop (after sorting).

```
38 For each offset (real number)  $0 \leq t \leq 1$ :
39 {
40    $x(t) = (EndPoint_x - StartPoint_x) * t + StartPoint_x$ 
41    $y(t) = (EndPoint_y - StartPoint_y) * t + StartPoint_y$ 
42    $c(t) = BLEND((t - t_{lo}) / (t_{hi} - t_{lo}), c_{lo}, c_{hi})$ 
43    $a(t) = [(t - t_{lo}) / (t_{hi} - t_{lo})] * (a_{hi} - a_{lo}) + a_{lo}$ 
44 }
```

45 Where  $t_{lo}$  and  $t_{hi}$  are the offsets,  $c_{lo}$  and  $c_{hi}$  are the color component values at the  
 46 closest enclosing gradient stops (that is,  $t_{lo} \leq t \leq t_{hi}$ ) and  $a_{lo}$  and  $a_{hi}$  are the alpha  
 47 component values at the closest enclosing gradient stops ( $t_{lo} \leq t \leq t_{hi}$ ).

```

1   For each offset (real number)  $t > 1$ :
2   {
3        $x(t) = (EndPoint_x - StartPoint_x) * t + StartPoint_x$ 
4        $y(t) = (EndPoint_y - StartPoint_y) * t + StartPoint_y$ 
5        $c(t) = c_{last}$ 
6        $a(t) = a_{last}$ 
7   }

```

8 Where  $c_{last}$  are the color component values of the last gradient stop (after sorting) and  
9  $a_{last}$  is the alpha component value at the last gradient stop (after sorting).

- 10 3. If the SpreadMethod value is Repeat, the colors of points on the line defined by the  
11 StartPoint and EndPoint attributes are defined by interpolating the coordinates linearly, and  
12 each color component (such as R, G, B for sRGB and scRGB) as well as the alpha  
13 component is interpolated between the component values of the closest enclosing  
14 gradient stops:

```

15 For each repetition (all integers) N:
16 {
17     For each offset (real number)  $0 \leq t < 1$ :
18     {
19          $x(t) = (EndPoint_x - StartPoint_x) * (N+t) + StartPoint_x$ 
20          $y(t) = (EndPoint_y - StartPoint_y) * (N+t) + StartPoint_y$ 
21          $c(t) = BLEND((t - t_{lo}) / (t_{hi} - t_{lo}), c_{lo}, c_{hi})$ 
22          $a(t) = [(t - t_{lo}) / (t_{hi} - t_{lo})] * (a_{hi} - a_{lo}) + a_{lo}$ 
23     }
24 }

```

25 Where  $c$  is the color component and  $a$  is the alpha component.  $t_{lo}$  and  $t_{hi}$  are the offsets,  
26  $c_{lo}$  and  $c_{hi}$  are the color component values at the closest enclosing gradient stops (that is,  
27  $t_{lo} \leq t \leq t_{hi}$ ) and  $a_{lo}$  and  $a_{hi}$  are the alpha component values at the closest enclosing  
28 gradient stops ( $t_{lo} \leq t \leq t_{hi}$ ).

- 29 4. If the SpreadMethod value is Reflect, the colors of points on the line defined by the  
30 StartPoint and EndPoint attributes are defined by interpolating the coordinates linearly, and  
31 each color component (such as R, G, B for sRGB and scRGB) as well as the alpha  
32 component is interpolated between the component values of the closest enclosing  
33 gradient stops:

```

34 For each repetition (all integers) N:
35 {
36     For each offset (real number)  $0 \leq t \leq 1$ :
37     {
38         If (N is EVEN)
39         {
40              $x(t) = (EndPoint_x - StartPoint_x) * (N+t) + StartPoint_x$ 
41              $y(t) = (EndPoint_y - StartPoint_y) * (N+t) + StartPoint_y$ 
42         }
43         Else
44         {
45              $x(t) = (EndPoint_x - StartPoint_x) * (N+1-t) + StartPoint_x$ 
46              $y(t) = (EndPoint_y - StartPoint_y) * (N+1-t) + StartPoint_y$ 
47         }
48          $c(t) = BLEND((t - t_{lo}) / (t_{hi} - t_{lo}), c_{lo}, c_{hi})$ 
49          $a(t) = [(t - t_{lo}) / (t_{hi} - t_{lo})] * (a_{hi} - a_{lo}) + a_{lo}$ 
50     }
51 }

```



- 1       Where  $c$  is the color component and  $a$  is the alpha component.  $t_{lo}$  and  $t_{hi}$  are the offsets,  
 2        $c_{lo}$  and  $c_{hi}$  are the color component values at the closest enclosing gradient stops (that is,  
 3        $t_{lo} \leq t \leq t_{hi}$ ) and  $a_{lo}$  and  $a_{hi}$  are the alpha component values at the closest enclosing  
 4       gradient stops ( $t_{lo} \leq t \leq t_{hi}$ ).
- 5       5. The colors of points not on the extended line defined by the StartPoint and EndPoint  
 6       attributes are the same as the color of the closest point on the line defined by the  
 7       StartPoint and EndPoint attributes, measured in the coordinate space as transformed by  
 8       the current effective render transform (including the render transform for the element  
 9       being filled by the linear gradient brush and the brush's transform itself).
- 10      6. Clip the resulting set of points to the intersection of the current clip geometry and the  
 11      path or glyphs to be filled. Both the clip and path (or glyphs) must be transformed  
 12      according to the current effective render transform, including the render transform for  
 13      the element being filled, but *not* including the transform of the linear gradient brush.

14      For purposes of the above steps, the closest enclosing gradient stops mean the gradient stops  
 15      that, if the relative sequencing of the gradient stop offsets in the markup order is respected,  
 16      are numerically closest to the interpolation point if that interpolation point were converted to an  
 17      offset value and inserted in a sorted fashion into the list of gradient stops. [Example: If a  
 18      gradient contains three gradient stops at offset values 0.0, 0.0, and 1.0, the closest enclosing  
 19      gradient stops for any value  $0 \leq \text{value} \leq 1$  are the second gradient stop (offset 0.0) and the  
 20      third gradient stop (offset 1.0). *end example*]

### 21   18.3.3 Radial Gradients

22      Consumers SHOULD render an element filled with a radial gradient brush such that the  
 23      appearance is the same as if these steps had been followed [S11.34]:

- 24      1. The boundary of the area filled by a radial gradient brush is defined by interpolating  
 25      ellipses from the GradientOrigin value to the circumference of the ellipse centered at the  
 26      point specified by the Center attribute with radii equal to the RadiusX and RadiusY  
 27      attribute values (the interpolated ellipses and point being transformed by the current  
 28      effective render transform, including the render transform for the element being filled by  
 29      the radial gradient brush and the brush's transform itself). If the gradient origin is  
 30      outside the circumference of the ellipse specified, the effect will be as if a cone were  
 31      drawn, tapering to the gradient origin.
- 32      2. If the SpreadMethod value is Pad, the centers and radii of the interpolated ellipses are  
 33      defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute  
 34      value to the Center attribute value, and simultaneously linearly interpolating the radii of  
 35      the ellipse from 0 to the RadiusX and RadiusY attribute values:

36      For each offset (real number)  $0 \leq t \leq 1$ :

```

37      {
38           $c_x(t) = (\text{Center}_x - \text{GradientOrigin}_x) * t + \text{GradientOrigin}_x$ 
39           $c_y(t) = (\text{Center}_y - \text{GradientOrigin}_y) * t + \text{GradientOrigin}_y$ 
40           $r_x(t) = \text{RadiusX} * t$ 
41           $r_y(t) = \text{RadiusY} * t$ 
42      }
```

43      The ellipses defined by the interpolation are transformed by the current effective render  
 44      transform, including the render transform for the element being filled by the radial  
 45      gradient brush and the brush's transform itself.

- 46      3. The colors of the points within the boundary of this shape are defined as the color of the  
 47      smallest interpolated ellipse containing the point. The color of an interpolated ellipse is

1 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as  
 2 well as the alpha component between the component values of the closest enclosing  
 3 gradient stops:

```
4 For each offset (real number)  $0 \leq t \leq 1$ :
5 {
6    $c(t) = \text{BLEND}((t-t_{10})/(t_{hi}-t_{10}), c_{10}, c_{hi})$ 
7    $a(t) = [(t-t_{10})/(t_{hi}-t_{10})]*(a_{hi}-a_{10})+a_{10}$ 
8 }
```

9 Where  $t_{10}$  and  $t_{hi}$  are the offsets,  $c_{10}$  and  $c_{hi}$  are the color component values at the  
 10 closest enclosing gradient stops (that is,  $t_{10} \leq t \leq t_{hi}$ ) and  $a_{10}$  and  $a_{hi}$  are the alpha  
 11 component values at the closest enclosing gradient stops ( $t_{10} \leq t \leq t_{hi}$ ).

12 4. If the SpreadMethod value is Repeat, the centers and radii of the interpolated ellipses are  
 13 defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute  
 14 value to the Center attribute value, and simultaneously linearly interpolating the radii of  
 15 the ellipse from 0 to RadiusX and RadiusY attribute values:

```
16 For each repetition (all non-negative integers) N:
17 {
18   For each offset (real number)  $0 \leq t < 1$ :
19   {
20      $c_x(t) = (\text{Center}_x - \text{GradientOrigin}_x)*(N+t) + \text{GradientOrigin}_x$ 
21      $c_y(t) = (\text{Center}_y - \text{GradientOrigin}_y)*(N+t) + \text{GradientOrigin}_y$ 
22      $r_x(t) = \text{RadiusX}*(N + t)$ 
23      $r_y(t) = \text{RadiusY}*(N + t)$ 
24   }
25 }
```

26 The ellipses defined by the interpolation are transformed by the current effective render  
 27 transform, including the render transform for the element being filled by the radial  
 28 gradient brush and the brush's transform itself.

29 5. The colors of the points within the boundary of this shape are defined as the color of the  
 30 smallest interpolated ellipse containing the point. The color of an interpolated ellipse is  
 31 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as  
 32 well as the alpha component between the component values of the closest enclosing  
 33 gradient stops:

```
34 For each repetition (all non-negative integers) N:
35 {
36   For each offset (real number)  $0 \leq t < 1$ :
37   {
38      $c(t) = \text{BLEND}((t-t_{10})/(t_{hi}-t_{10}), c_{10}, c_{hi})$ 
39      $a(t) = [(t-t_{10})/(t_{hi}-t_{10})]*(a_{hi}-a_{10})+a_{10}$ 
40   }
41 }
```

42 Where  $t_{10}$  and  $t_{hi}$  are the offsets,  $c_{10}$  and  $c_{hi}$  are the color component values at the  
 43 closest enclosing gradient stops (that is,  $t_{10} \leq t \leq t_{hi}$ ) and  $a_{10}$  and  $a_{hi}$  are the alpha  
 44 component values at the closest enclosing gradient stops ( $t_{10} \leq t \leq t_{hi}$ ).

45 6. If the SpreadMethod value is Reflect, the centers and radii of the interpolated ellipses are  
 46 defined by linearly interpolating the center of the ellipse from the GradientOrigin attribute  
 47 value to the Center attribute value, and simultaneously linearly interpolating the radii of  
 48 the ellipse from 0 to the RadiusX and RadiusY attribute values:

```

1   For each non-negative integer N:
2   {
3     For each offset (real number)  $0 \leq t \leq 1$ :
4     {
5        $c_x(t) = (\text{Center}_x - \text{GradientOrigin}_x) * (N+t) + \text{GradientOrigin}_x$ 
6        $c_y(t) = (\text{Center}_y - \text{GradientOrigin}_y) * (N+t) + \text{GradientOrigin}_y$ 
7        $r_x(t) = \text{RadiusX} * (N+t)$ 
8        $r_y(t) = \text{RadiusY} * (N+t)$ 
9     }
10  }

```

11 The ellipses defined by the interpolation are transformed by the current effective render  
12 transform, including the render transform for the element being filled by the radial  
13 gradient brush and the brush's transform itself.

14 7. The colors of the points within the boundary of this shape are defined as the color of the  
15 smallest interpolated ellipse containing the point. The color of an interpolated ellipse is  
16 defined by interpolating each color component (such as R, G, B for sRGB and scRGB) as  
17 well as the alpha component between the component values of the closest enclosing  
18 gradient stops:

```

19  For each non-negative integer N:
20  {
21    For each offset (real number)  $0 \leq t \leq 1$ :
22    {
23      If N is ODD
24         $t' = 1-t$ 
25      Else
26         $t' = t$ 
27
28       $c(t) = \text{BLEND}((t' - t_{1o}) / (t_{hi} - t_{1o}), c_{1o}, c_{hi})$ 
29       $a(t) = [(t' - t_{1o}) / (t_{hi} - t_{1o})] * (a_{hi} - a_{1o}) + a_{1o}$ 
30    }
31  }

```

32 Where  $t_{1o}$  and  $t_{hi}$  are the offsets,  $c_{1o}$  and  $c_{hi}$  are the color component values at the  
33 closest enclosing gradient stops (that is,  $t_{1o} \leq t \leq t_{hi}$ ) and  $a_{1o}$  and  $a_{hi}$  are the alpha  
34 component values at the closest enclosing gradient stops ( $t_{1o} \leq t \leq t_{hi}$ ).

35 8. The colors of points outside the boundary of this shape (points which cannot be drawn by  
36 any combination of non-negative N and t) are defined as having the color and alpha  
37 defined in the gradient stop with the offset of 0.0 for radial gradients with a SpreadMethod  
38 value of Reflect and the color and alpha defined in the gradient stop with the offset of 1.0  
39 for radial gradients with a SpreadMethod value of Repeat or Pad. The colors outside of the  
40 boundary of this shape can also vary in an implementation-defined manner  
41 (see §18.3.1.2 for more details).

42 9. Clip the resulting set of points by the intersection of the current clip geometry and the  
43 path or glyphs to be filled. Both the clip and path (or glyphs) must be transformed  
44 according to the current effective render transform, including the render transform for  
45 the element being filled, but *not* including the transform of the radial gradient brush.

46 For purposes of the above steps, the closest enclosing gradient stops mean the gradient stops  
47 that, if the relative sequencing of the gradient stop offsets in the markup order is respected,  
48 are numerically closest to the interpolation point if that interpolation point were converted to an  
49 offset value and inserted in a sorted fashion into the list of gradient stops. [Example: If a  
50 gradient contains three gradient stops at offset values 0.0, 0.0, and 1.0, the closest enclosing

1 gradient stops for any value  $0 \leq \text{value} \leq 1$  are the second gradient stop (offset 0.0) and the  
 2 third gradient stop (offset 1.0). *end example*]

### 3 **18.4 Opacity Computations**

4 Opacity is used to transparently blend two elements when rendering, also known as alpha  
 5 blending. The value of the Opacity property ranges from 0.0 (fully transparent) to 1.0 (fully  
 6 opaque), inclusive. Values outside of this range are invalid.

7 The opacity is applied through the following computations, assuming source and destination  
 8 values are not pre-multiplied. All opacity calculations SHOULD be performed with at least 8-bit  
 9 precision to provide sufficient quality for nested content [S11.18].

10 Individual pixels are blended as defined below.

11 *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
$A_R$	Resulting Alpha value for destination surface
$R_R$	Resulting Red value for destination surface
$G_R$	Resulting Green value for destination surface
$B_R$	Resulting Blue value for destination surface

12 All values designated with a  $T$  subscript (as in  $R_{T1}$ ) are temporary values.

13 The opacity is calculated as follows:

14 1. Multiply source alpha value with opacity value and alpha value of opacity mask.

15 
$$A_{S1} = A_S * O_E * O_M$$

16 2. Pre-multiply source alpha.

17 If the source data specifies pre-multiplied alpha (see §18.4.1 for details)  $A_{T1} = 0$ ,  $R_{T1} = R_S$ ,  
 18  $G_{T1} = G_S$ ,  $B_{T1} = B_S$ ; otherwise:

19 
$$A_{T1} = A_{S1}$$

20 
$$R_{T1} = R_S * A_{S1}$$

21 
$$G_{T1} = G_S * A_{S1}$$

22 
$$B_{T1} = B_S * A_{S1}$$

## 1        3. Pre-multiply destination alpha.

2        If a consumer supports superluminous colors (see §18.4.1 for details)  $A_{T2} = A_D$ ,  $R_{T2} = R_D$ ,  
 3         $G_{T2} = G_D$ ,  $B_{T2} = B_D$ ; otherwise:

4         $A_{T2} = A_D$   
 5         $R_{T2} = R_D * A_D$   
 6         $G_{T2} = G_D * A_D$   
 7         $B_{T2} = B_D * A_D$

## 8        4. Blend.

9        See §18.4.1 for special case handling.

10        $A_{T3} = (1 - A_{T1}) * A_{T2} + A_{T1}$   
 11        $R_{T3} = (1 - A_{T1}) * R_{T2} + R_{T1}$   
 12        $G_{T3} = (1 - A_{T1}) * G_{T2} + G_{T1}$   
 13        $B_{T3} = (1 - A_{T1}) * B_{T2} + B_{T1}$

## 14       5. Reverse pre-multiplication.

15       The resulting color channel values are divided by the resulting alpha value. If the  
 16       resulting alpha value is 0, all color channels are set to 0 by definition, as expressed in the  
 17       If condition below. Each of  $R_{T3}$ ,  $G_{T3}$ ,  $B_{T3}$  is smaller than or equal to  $A_{T3}$  and, therefore, each  
 18       of the resulting  $R_R$ ,  $G_R$ ,  $B_R$  is in the valid interval of [0.0,1.0] after the pre-multiplication is  
 19       reversed.

20       If a consumer supports superluminous colors

21       {  
 22            $A_R = A_{T3}$ ,    $R_R = R_{T3}$ ,    $G_R = G_{T3}$ ,    $B_R = B_{T3}$   
 23       }

24       Else If  $A_{T3} = 0$

25       {  
 26           set all  $A_R$   $R_R$   $G_R$   $B_R$  to 0.  
 27       }

28       Else

29       {  
 30            $A_R = A_{T3}$   
 31            $R_R = R_{T3}/A_{T3}$   
 32            $G_R = G_{T3}/A_{T3}$   
 33            $B_R = B_{T3}/A_{T3}$   
 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.  
 37       Colors in subtractive color spaces (such as CMYK) are complemented before and after the  
 38       blending steps described above.

### 18.4.1 Pre-Multiplied Alpha and Superluminous Colors

The alpha information in TIFF images using an ExtraSamples tag value of 1 and in [Windows Media Photo JPEG XR](#) images using pixel formats [WICPixelFormat32bppPBGRA](#), [WICPixelFormat64bppPRGBA](#) or [WICPixelFormat128bppPRGBAFloat](#) MUST be interpreted as pre-multiplied alpha information [M11.6]. In certain scenarios (such as when rendering 3D scenes to a bitmap), producers MAY choose to create pre-multiplied bitmap data specifying “superluminous” colors [O11.9].

Superluminous colors are defined as a subset case of the pre-multiplied RGB source color values case in which the source alpha value is smaller than the individual color channel values but greater than or equal to 0.

The effect of composing superluminous colors on a background is similar to adding additional light of the source color to the destination color, as opposed to regular alpha composition which works more like a colored filter. One can easily verify this statement by substituting 0 for  $A_{T1}$  in step 4 of the above opacity computations, which is simplified as follows:

$$\begin{aligned} A_{T3} &= A_{T2} \\ R_{T3} &= R_{T2} + R_{T1} \\ G_{T3} &= G_{T2} + G_{T1} \\ B_{T3} &= B_{T2} + B_{T1} \end{aligned}$$

Consumers supporting superluminous colors retain all temporary information in pre-multiplied formats. Note, that throughout the OpenXPS Standard non-pre-multiplied alpha processing is assumed. It is up to the implementer of such a consumer to identify equivalent composition and rendering rules for processing in pre-multiplied space.

Also note, 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 [S11.19].

Consumers MAY handle superluminous colors or MAY instead choose to convert pre-multiplied source data containing superluminous colors to non-pre-multiplied data before composition by ignoring the superluminous portion of each color channel value [O11.10], as described in the following steps:

```

For each superluminous pixel with  $A_S < R_S$  or  $A_S < G_S$  or  $A_S < B_S$ 
{
  If  $A_S = 0$ 
  {
     $A_R = 0$ 
     $R_R = 1$ 
     $G_R = 1$ 
     $B_R = 1$ 
  }
  Else
  {
     $A_R = A_S$ 
     $R_R = \min(R_S/A_S, 1)$ 
     $G_R = \min(G_S/A_S, 1)$ 
     $B_R = \min(B_S/A_S, 1)$ 
  }
}

```

---

## 18.5 Composition Rules

OpenXPS 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 1. 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 2. If an element is fully transparent (an opacity of 0.0), it MAY be skipped [O11.13].
- 38 3. 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 4. 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 5. 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       6. 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**

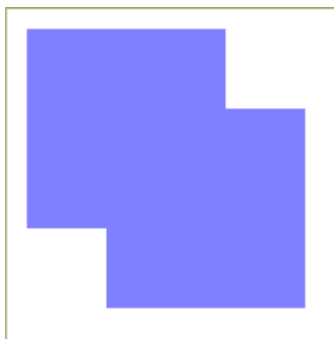
5   The following examples illustrate the composition rules described above.

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.

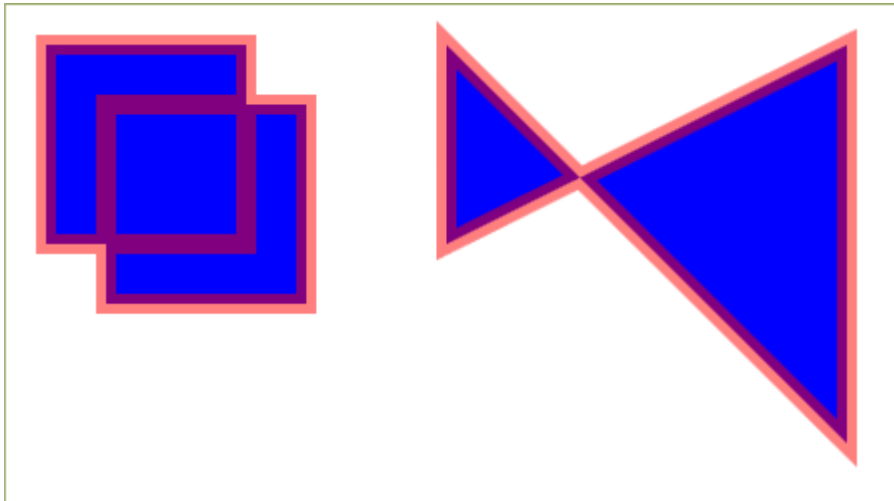
```
30     <Path Stroke="#80FF0000" StrokeThickness="10">
31       <Path.Fill>
32           <SolidColorBrush Color="#0000FF" />
33       </Path.Fill>
34       <Path.Data>
35           <PathGeometry FillRule="NonZero">
36               <PathFigure StartPoint="20,20" IsClosed="true">
```

```

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>

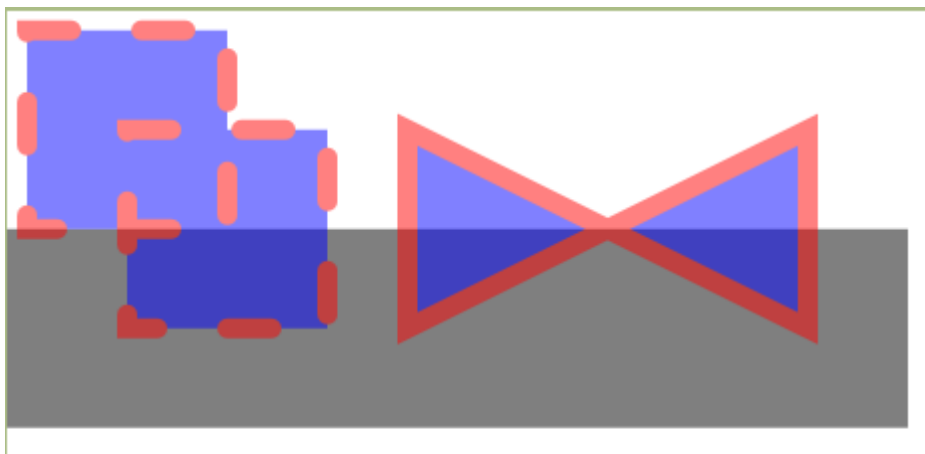
```

```

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**

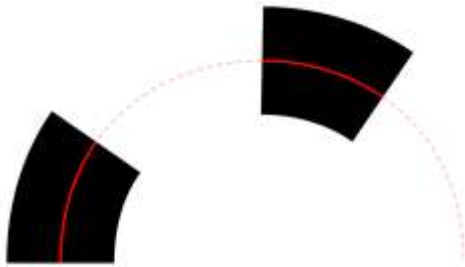
2 The appearance of dash caps is controlled by the `StrokeDashCap` attribute. Valid values are `Flat`,  
3 `Square`, `Round`, and `Triangle`. The `StrokeDashCap` attribute is ignored for paths that have no  
4 `StrokeDashArray` attribute or that have a `StrokeDashArray` attribute with value `0,0`.

##### 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

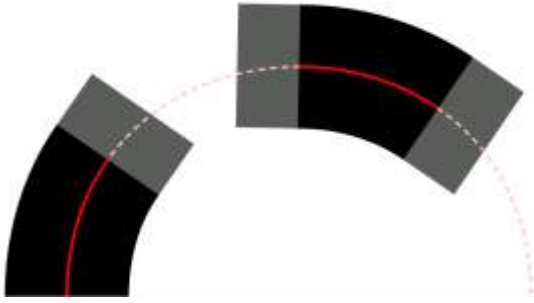
#### 14 **18.6.4.2 Square Dash Caps**

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 square  
19 caps attached) and the contour of the shape.

20 The caps are drawn as half-squares attached to the ends of the dash. The boundaries of the  
21 square caps are not curved to follow the contour, but are transformed using the effective  
22 render transform.

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 squares. If a dash with a  
25 length of 0 appears at, or very near to, a join in a path then differences in rendering resolution  
26 and in precision in the calculation of coordinates may lead to differing orientation of the dash  
27 caps between consumers.

1 *Figure 18-3. Square dash caps*

2

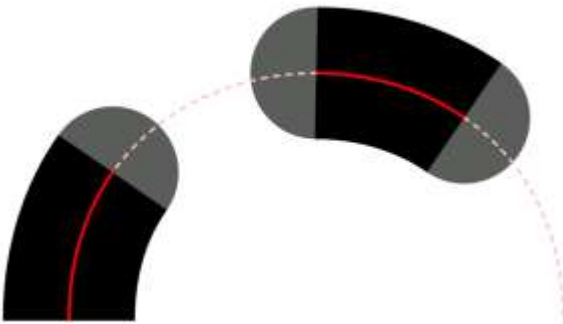
3 **18.6.4.3 Round Dash Caps**

4 The effective render transform of the path being stroked is used to transform the control points  
5 of the contour of the dash.

6 The length of the dash is the approximate distance on the curve between the two contour  
7 intersection points, that is, the intersection of the flat line ending the dash (without the round  
8 caps attached) and the contour of the shape.

9 The caps are drawn as half-circles attached to the ends of the dash. The boundaries of the  
10 round caps are not distorted to follow the contour, but are transformed using the effective  
11 render transform.

12 The distance between the contour intersection points of consecutive dashes is the specified  
13 dash gap length. Degenerate dashes with a length of 0 are drawn as circles.

14 *Figure 18-4. Round dash caps*

15

16 **18.6.4.4 Triangular Dash Caps**

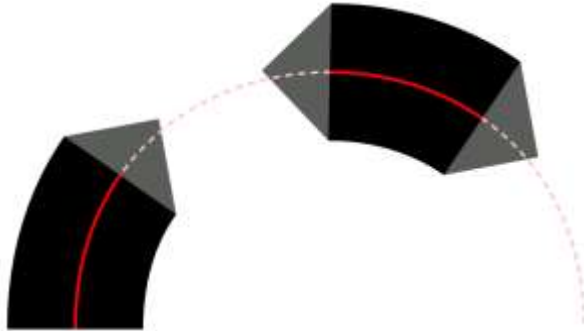
17 The effective render transform of the path being stroked is used to transform the control points  
18 of the contour of the dash.

19 The length of the dash is the approximate distance on the curve between the two contour  
20 intersection points, that is, the intersection of the flat line ending the dash (without the  
21 triangular caps attached) and the contour of the shape.

22 The caps are drawn as triangles attached with their base to the ends of the dash. The  
23 boundaries of the triangular caps are not distorted to follow the contour, but are transformed  
24 using the effective render transform. The height of the triangles is half of the stroke width.

1 The distance between the contour intersection points of consecutive dashes is the specified  
 2 dash gap length. Degenerate dashes with a length of 0 are drawn as diamonds. If a dash with a  
 3 length of 0 appears at, or very near to, a join in a path then differences in rendering resolution  
 4 and in precision in the calculation of coordinates may lead to differing orientation of the dash  
 5 caps between consumers.

6 *Figure 18-5. Triangular dash caps*

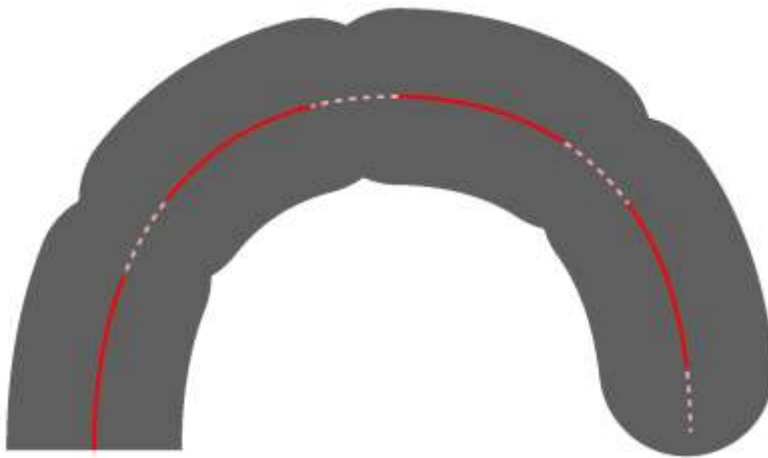


7

#### 8 **18.6.4.5 Overlapping Dashes**

9 It is possible to specify dash sequences with overlapping dash caps. In this circumstance, the  
 10 union of the dash segments (inclusive of dash caps), is used as a mask through which the  
 11 brush is applied as illustrated in Figure 18-6 with a stroke dash cap value of Round.

12 *Figure 18-6. Overlapping dash segments*



13

#### 14 **18.6.4.6 Extreme Degenerate Dash Case**

15 The previous subclauses include a description of the behaviour for degenerate dashes of zero  
 16 length, with non-zero gaps, for each dash cap shape.

17 Producers SHOULD NOT create files containing the extreme degenerate case of  
 18 `StrokeDashArray = "0 0"`. Such lines SHOULD be rendered as a solid line [S11.32].

#### 19 **18.6.5 Rules for Line Cap Rendering**

20 The appearance of line caps is controlled by the `StrokeStartLineCap` and `StrokeEndLineCap`  
 21 attribute. Valid values are Flat, Square, Triangle, and Round. Every start line cap can be used in

1 combination with any end line cap. Line caps only ever appear at the start and end of an open  
 2 path, and then only if the initial/final segment is stroked.

3 The rules for line caps on curved lines are analogous to the rules for dash cap rendering. For  
 4 more information, see §18.6 and §18.6.4.

5 *Figure 18-7. Flat start line cap, flat end line cap*



6

7 *Figure 18-8. Square start line cap, square end line cap*



8

9 *Figure 18-9. Triangular start line cap, triangular end line cap*



10

11 *Figure 18-10. Round start line cap, round end line cap*



12

### 13 **18.6.6 Line Caps for Dashed Strokes**

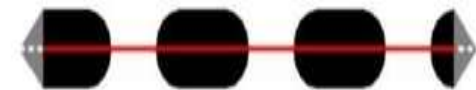
14 If the start point of a stroke is within a dash or touches the start or end of a dash, a start line  
 15 cap is appended to the stroke. Similarly, if the end point of a stroke is within a dash or touches  
 16 the start or end of a dash, an end line cap is appended to the stroke.

17 *Figure 18-11. Stroke start or end point within a dash for flat dash caps*



18

19 *Figure 18-12. Stroke start or end point within a dash for non-flat dash caps*



20

21 [*Note*: Because the right-most line cap begins at the point exactly coincident with the start of  
 22 the next dash in the sequence, it is rendered. *end note*]



- 1 However, if the start point of a stroke is within a gap (as can result from a StrokeDashOffset  
 2 attribute), no start line cap is appended to the stroke. If the end point of a stroke is within a  
 3 gap, no end line cap is appended to the stroke.

4 *Figure 18–13. Stroke start or end point within a gap for flat dash caps*



5

6 *Figure 18–14. Stroke start or end point within a gap for not-flat dash caps*



7

- 8 [Note: Differences in precision in the calculation of coordinates can lead to differing output  
 9 between consumers depending on whether they determine that the start or end point of a  
 10 stroke exactly touches the start or end point of a dash. *end note*]

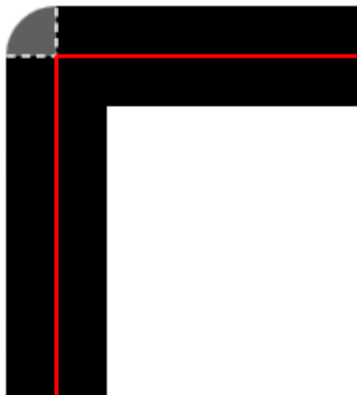
### 11 **18.6.7 Rules for Line Join Rendering**

12 The appearance of line joins is controlled by the StrokeLineJoin attribute. Valid values are Round,  
 13 Bevel, and Miter.

#### 14 **18.6.7.1 Round Line Joins**

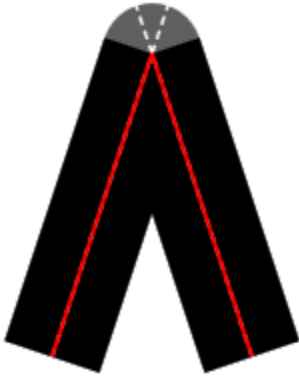
15 A StrokeLineJoin attribute value of Round indicates that the outer corner of the joined lines  
 16 should be filled by enclosing the rounded region with its center point at the point of intersection  
 17 between the two lines and a radius of one-half the stroke thickness value.

18 *Figure 18–15. Round line join with right angle*



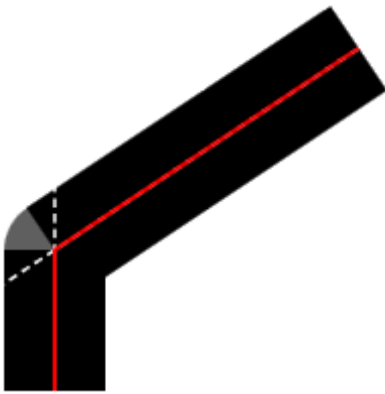
19

1 *Figure 18-16. Round line join with acute angle*



2

3 *Figure 18-17. Round line join with obtuse angle*

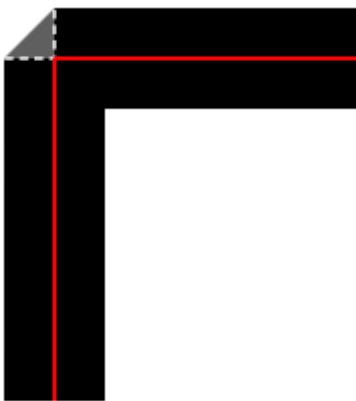


4

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*



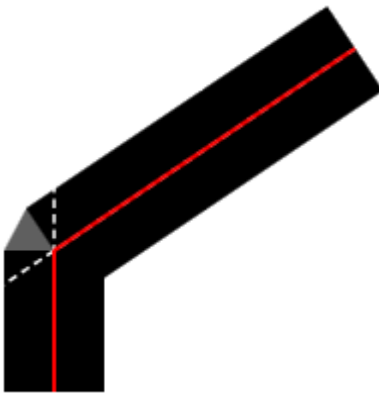
10

1 *Figure 18–19. Beveled line join with acute angle*



2

3 *Figure 18–20. Beveled line join with obtuse angle*



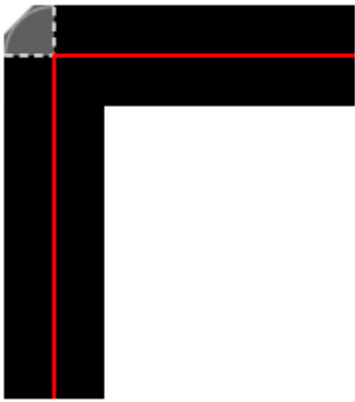
4

### 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.

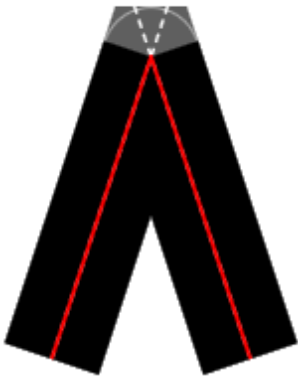
12 When drawing mitered line joins, the presence of one or more degenerate line segments  
 13 between the non-degenerate line segments to be joined results in a mitered line join of only the  
 14 two non-degenerate line segments with an implied `StrokeMiterLimit` attribute value of 1.0.

1 *Figure 18–21. Mitered line join with right angle and miter limit of 1.0*



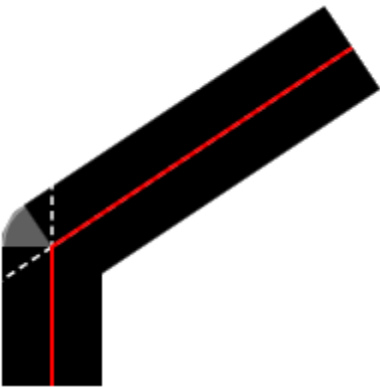
2

3 *Figure 18–22. Mitered line join with acute angle and miter limit of 1.0*



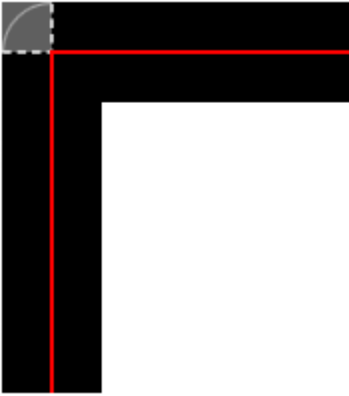
4

5 *Figure 18–23. Mitered line join with obtuse angle and miter limit of 1.0*



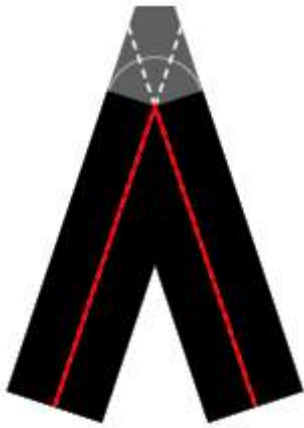
6

1 *Figure 18–24. Mitered line join with right angle and miter limit of 2.0*



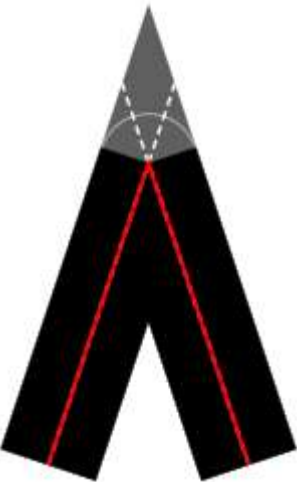
2

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.

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, [after application of relevant  
5 transforms](#), the same StrokeThickness attribute value, and consumers aware of this convention  
6 SHOULD render such a stroke no thinner than the thinnest visible line that [a bi-tonal](#) consumer  
7 supports without dropouts [or an anti-aliasing consumer can represent as a solid line. In the  
8 particular case of StrokeThickness attribute value of "0" the stroke SHOULD be rendered with a  
9 1-pixel thickness if the nominal stroke width convention applies](#) [S11.31]. See §18.1.4, for  
10 further considerations for rendering thin lines.

11 [\[Note : Producers using this convention must be aware that the rendering of thin and zero-  
12 width lines may have inconsistent results depending on the support or not of this convention at  
13 the consumer side. A line rendered as a solid line on a consumer supporting this convention  
14 may be rendered with different density or even not rendered at all on a consumer not  
15 supporting it. end note\]](#)

---

## 16 **18.7 Brushes and Images**

17 Images require the following special considerations for scaling and tile placement.

### 18 **18.7.1 Small Tiles**

19 Tiles for visual brushes and image brushes can be specified with a viewport width or height of a  
20 few device pixels, or even less than a single device pixel in size.

21 If both width and height are nearly zero, implementations SHOULD average the color values of  
22 the brush contents, resulting in a constant-color brush [S11.26]. *[Example:*

- 23 • A visual brush or image brush that contains a blue and white checkerboard pattern  
24 results in a solid light-blue fill as either the width or the height value approaches 0.0.
- 25 • A visual brush or image brush whose viewbox is constant-colored produces a constant-  
26 colored brush regardless of the width and height values of the viewport.

27 *end example]*

28 If only one of the width and height values is nearly zero, the brush should be constant-colored  
29 along lines parallel to the narrow side of the viewport. For cases such as these,  
30 implementations MAY differ [O11.21]. Producers SHOULD avoid producing such extreme cases  
31 and SHOULD NOT rely on any specific behavior when they do [S11.27].

### 32 **18.7.2 Image Scaling**

33 Source sampling SHOULD be done from the center of the pixel and should be mapped to the  
34 center of the pixel in the device-space [S11.28]. With one extent of the viewbox zero, sampling  
35 SHOULD be done along a line parallel to the non-zero side [S11.28]. With both extents of the  
36 viewbox zero, a point sample SHOULD be taken [S11.28].

37 When up-sampling an image presented at a lower resolution than the device resolution, bilinear  
38 filtering SHOULD be used [S11.29]. The precise source coordinates as specified by the viewbox  
39 MUST be used to place the up-sampled image tile, which is equivalent to using fractional pixels  
40 of the original source image [M11.8].

1 When down-sampling, at least a bilinear filter SHOULD be used [S11.30]. Consumers MAY  
2 choose to implement a more sophisticated algorithm, such as a Fant scaler, to prevent aliasing  
3 artifacts [O11.22].

#### 4 **18.7.3 Tile Placement**

5 Consumers MUST precisely position the tiles specified by the image brush and visual brush. If  
6 the specified values result in fractional device pixels, the consumer MUST calculate a running  
7 placement-error delta and adjust the placement of the next tile where the delta reaches a full  
8 device pixel in order to keep the tiles from being increasingly out of phase as the expanse of  
9 the path is filled [M11.9]. Consumers MAY choose any technique desired to achieve this  
10 requirement, such as linear filtering for seams, stretching of the tile (up-sampling or down-  
11 sampling), or pre-computing multiple tiles and adjusting behavior according to how the tiles fit  
12 on a grid [O11.23].

#### 13 **18.7.4 Tiling Transparent Visual Brushes and Image Brushes**

14 The contents of a visual brush's Visual property are first rendered to a temporary work canvas  
15 (according to the composition rules in §18.5.). The viewbox of the visual brush defines the tile  
16 or portion of the temporary canvas that is copied onto the specified geometry, stroke, or text.  
17 Likewise, an image specified by an image brush is also copied to a temporary work canvas. The  
18 viewbox also defines the tile for an image brush. In either case, the work canvas is scaled to  
19 properly match the edges of the tile to the size specified by the viewport.

20 Each pixel of the resultant tile is separately blended with the background of its destination,  
21 using the alpha of each pixel. This process is repeated for each tile replication, while respecting  
22 the TileMode attribute value, although the temporary work canvas MAY be re-used [O11.24].

23



# 19. Elements

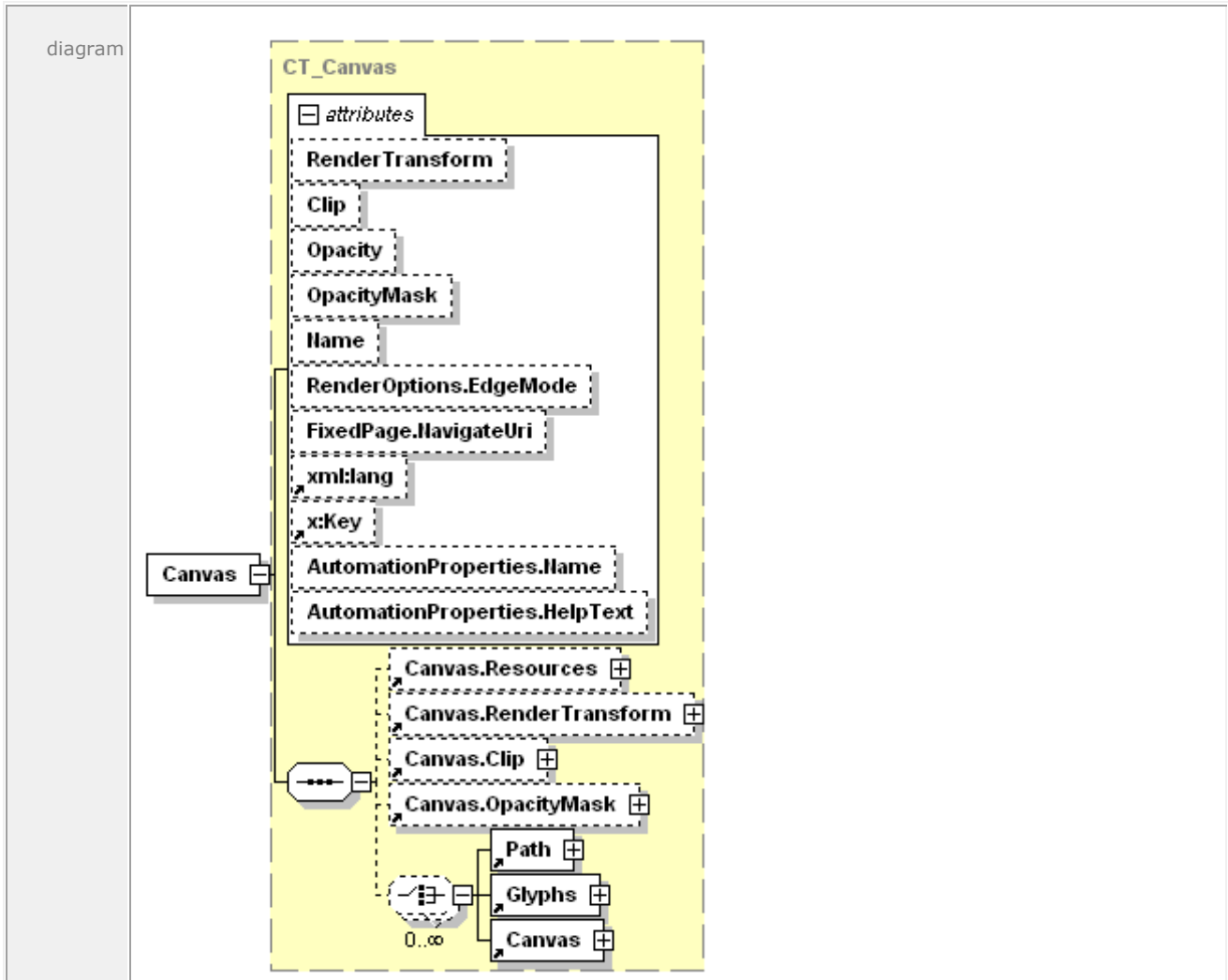
## 19.1 ArcSegment

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><a href="#">ST_Point</a></td> <td>required</td> <td></td> <td></td> <td>Specifies the endpoint of the elliptical arc.</td> </tr> <tr> <td>Size</td> <td><a href="#">ST_PointGE0</a></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><a href="#">ST_Double</a></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><a href="#">ST_Boolean</a></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><a href="#">ST_SweepDirection</a></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><a href="#">ST_Boolean</a></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	<a href="#">ST_Point</a>	required			Specifies the endpoint of the elliptical arc.	Size	<a href="#">ST_PointGE0</a>	required			Specifies the x and y radius of the elliptical arc as an x,y pair.	RotationAngle	<a href="#">ST_Double</a>	required			Indicates how the ellipse is rotated relative to the current coordinate system.	IsLargeArc	<a href="#">ST_Boolean</a>	required			Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.	SweepDirection	<a href="#">ST_SweepDirection</a>	required			Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.	IsStroked	<a href="#">ST_Boolean</a>		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	<a href="#">ST_Point</a>	required			Specifies the endpoint of the elliptical arc.																																						
Size	<a href="#">ST_PointGE0</a>	required			Specifies the x and y radius of the elliptical arc as an x,y pair.																																						
RotationAngle	<a href="#">ST_Double</a>	required			Indicates how the ellipse is rotated relative to the current coordinate system.																																						
IsLargeArc	<a href="#">ST_Boolean</a>	required			Determines whether the arc is drawn with a sweep of 180 or greater. Can be true or false.																																						
SweepDirection	<a href="#">ST_SweepDirection</a>	required			Specifies the direction in which the arc is drawn. Valid values are Clockwise and Counterclockwise.																																						
IsStroked	<a href="#">ST_Boolean</a>		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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				Limits the rendered region of the element.
	Opacity	<a href="#">ST_ZeroOne</a>		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 §14.3.1.

## 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 §14.4.2.

## 5 **19.6 Canvas.Resources**

6 element **Canvas.Resources**

diagram	
annotation	Contains the resource dictionary for the <Canvas> element.

7 For more information, see §14.2.2.

## 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**

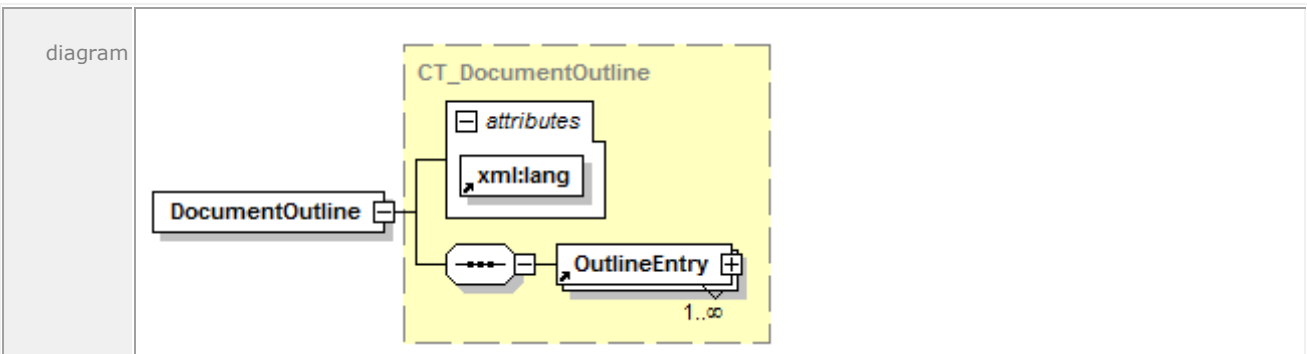


annotation	Contains a list of resources that are safe for a consumer to discard.
------------	---

4 For more information, see §17.1.4.1.1.

## 5 **19.9 DocumentOutline**

6 element **DocumentOutline**



attributes	Name	Type	Use	Default	Fixed	Annotation
	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.

annotation	Specifies a list of meaningful indices into the OpenXPS 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 OpenXPS 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.					

7  
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**

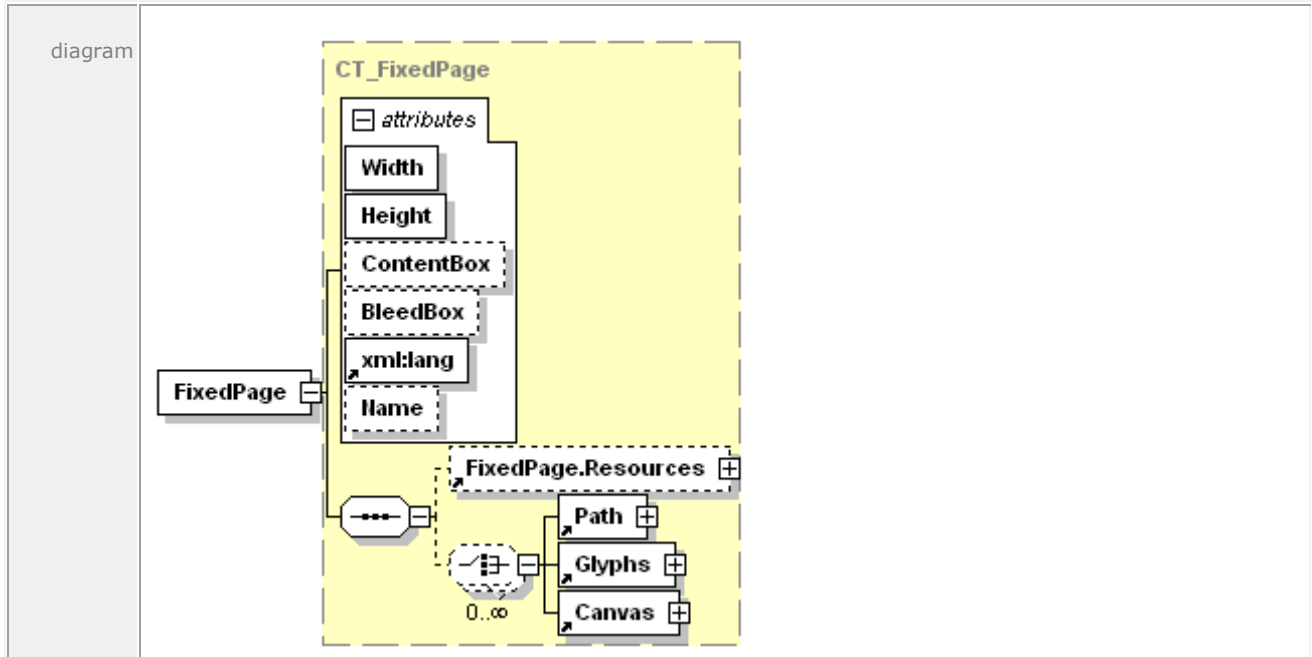
diagram	
annotation	Specifies a sequence of fixed documents.

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**

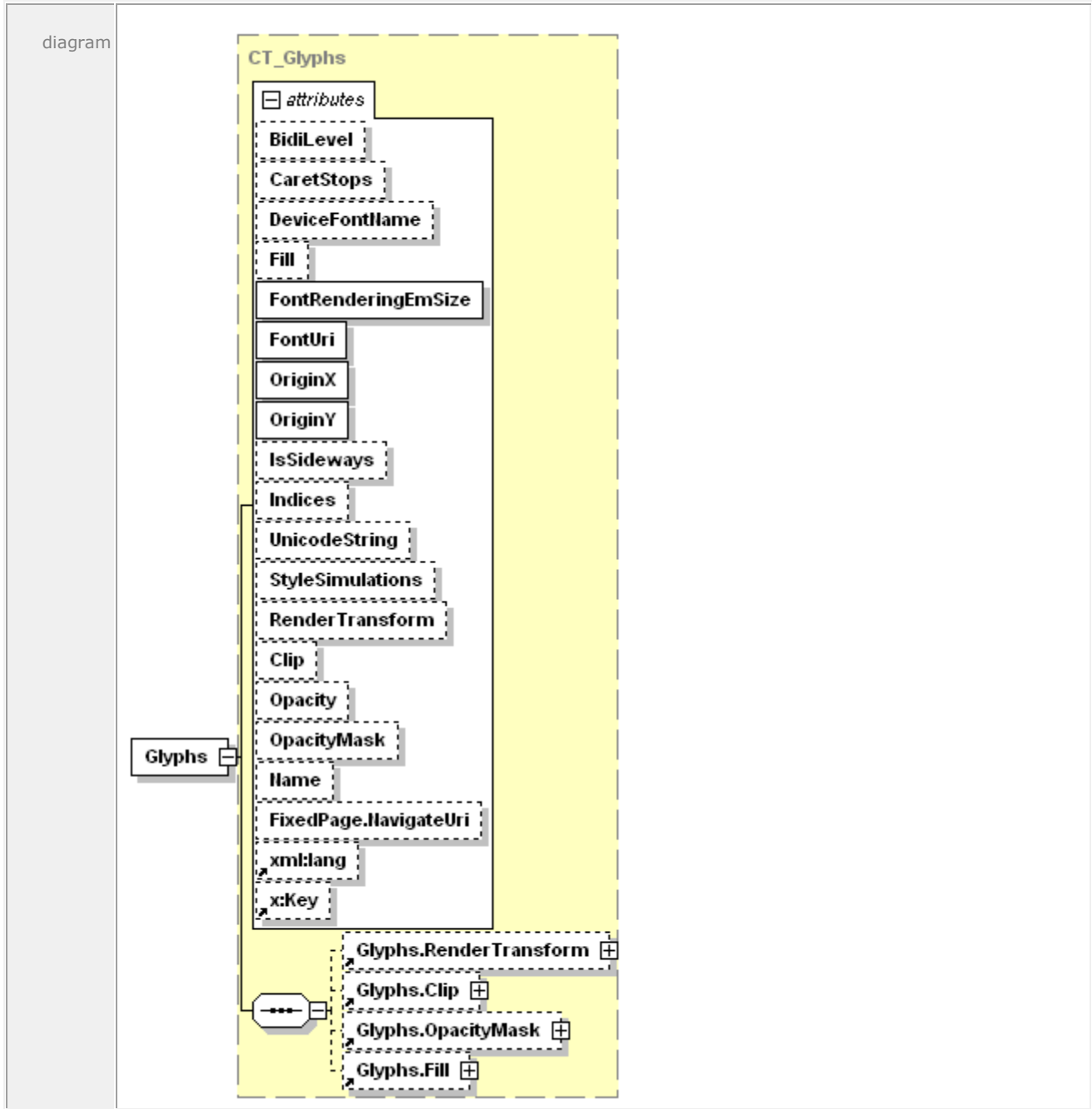
3 element **FixedPage.Resources**

diagram	<pre> classDiagram     class FixedPageResources["FixedPage.Resources"]     class ResourceDictionary     FixedPageResources ..&gt; ResourceDictionary     class CT_CP_Resources["CT_CP_Resources"]     ResourceDictionary .. &gt; CT_CP_Resources             </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. <a href="#">If the UnicodeString attribute of the &lt;Glyphs&gt; element is not specified or contains an empty value (" or "{}"), and if the Indices attribute is not specified or contains &lt;nothing&gt;, then a consumer MUST instantiate an error condition</a> <a href="#">If the UnicodeString attribute specifies an empty string (" or "{}") and the Indices attribute is not specified or is also empty, a consumer MUST instantiate an error condition</a> [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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				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	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRef</a>				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	<a href="#">ST_Name</a>				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 and §9.1.7.

## 2 19.19 Glyphs.Clip

3 element **Glyphs.Clip**

diagram	<pre> graph LR     Glyphs.Clip --&gt; PathGeometry     subgraph CT_CP_Geometry         PathGeometry     end             </pre>
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.

4 For more information, see §14.3.3.

## 5 19.20 Glyphs.Fill

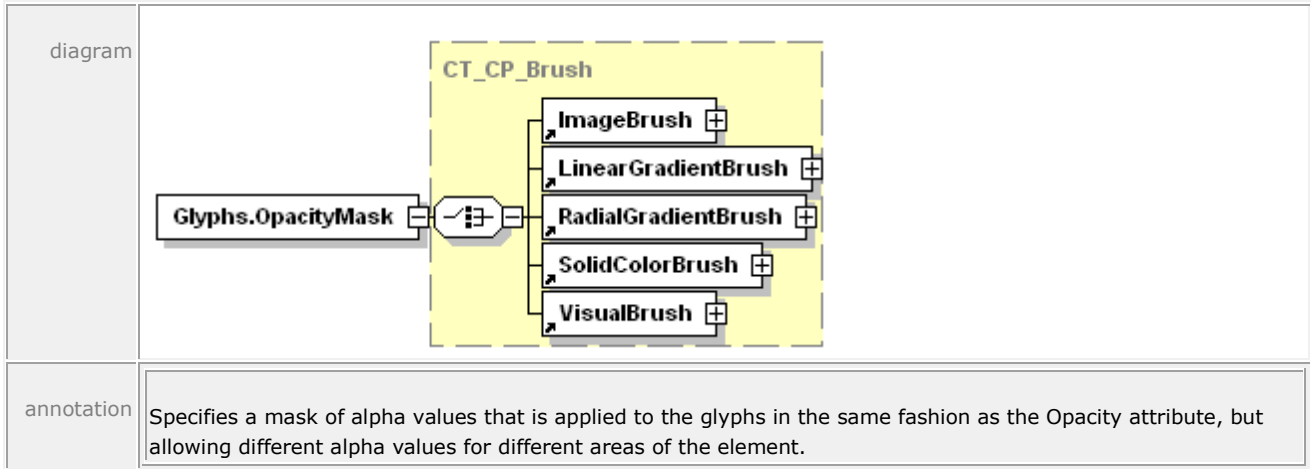
6 element **Glyphs.Fill**

diagram	<pre> graph LR     Glyphs.Fill --&gt; Node     subgraph CT_CP_Brush         Node --&gt; ImageBrush         Node --&gt; LinearGradientBrush         Node --&gt; RadialGradientBrush         Node --&gt; SolidColorBrush         Node --&gt; VisualBrush     end             </pre>
annotation	Describes the brush used to fill the shape of the rendered glyphs.

7 For more information, see §12.2.

## 8 19.21 Glyphs.OpacityMask

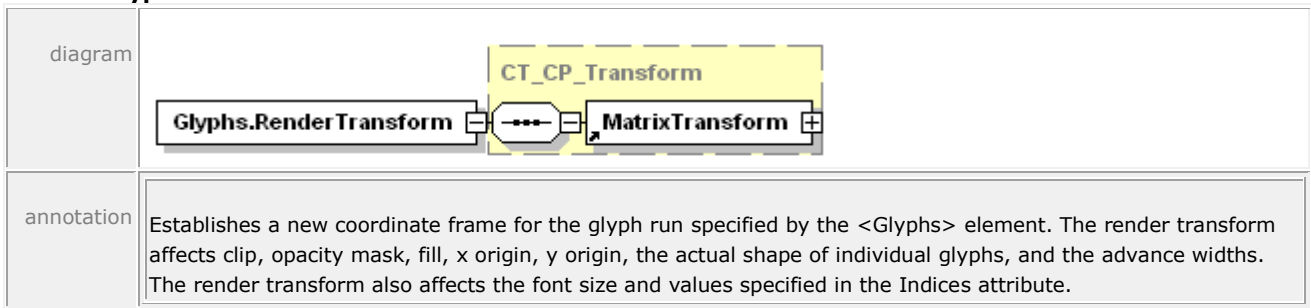
9 element **Glyphs.OpacityMask**



1 For more information, see §14.5.3.

## 2 19.22 Glyphs.RenderTransform

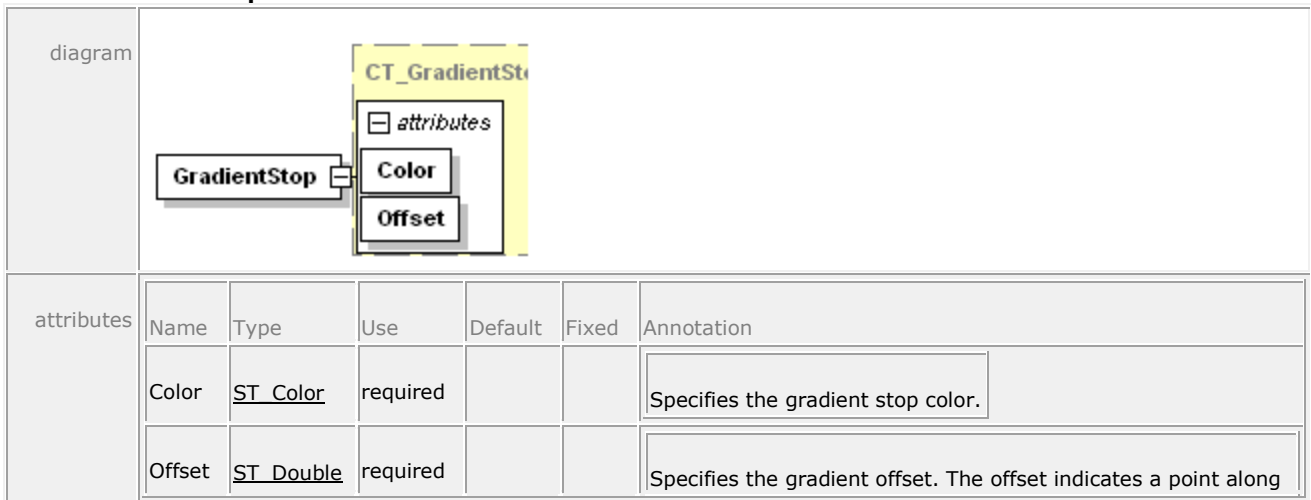
3 element **Glyphs.RenderTransform**



4 For more information, see §14.4.4.

## 5 19.23 GradientStop

6 element **GradientStop**



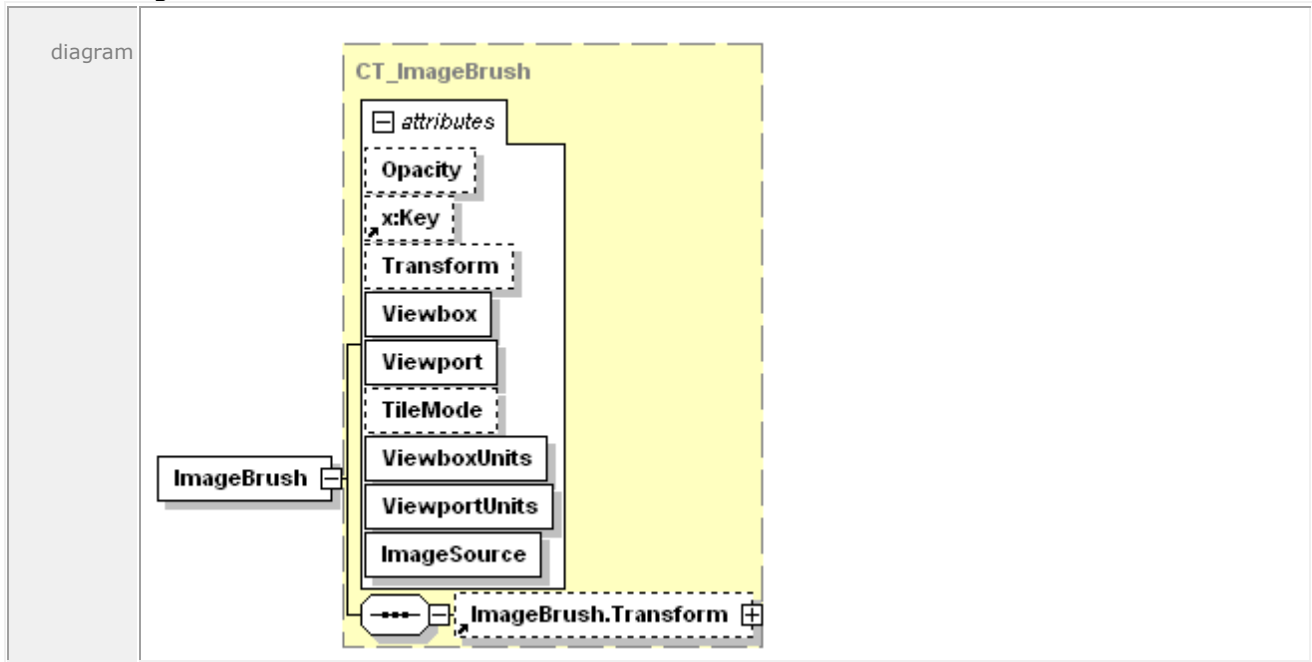


					the progression of the gradient at which a color is specified. Colors between gradient offsets in the progression are interpolated.
annotation	Indicates a location and range of color progression for rendering a gradient.				

1 For more information, see §13.7.

## 2 19.24 ImageBrush

3 element **ImageBrush**



attributes	Name	Type	Use	Default	Fixed	Annotation
		Opacity	<u>ST_ZeroOne</u>		1.0	
	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	<u>ST_RscRefMatrix</u>				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	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_TileMode</a>		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
ImageSource	<a href="#">ST_UriCtxBmp</a>	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	<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>

1 For more information, see §14.4.6.

## 2 19.26 Intent

3 element **SignatureDefinitionType/Intent**

diagram	
annotation	<p>A string that represents the intent to which the signing party agrees when signing the document.</p>

4 For more information, see §17.2.2.4.

## 5 19.27 LinearGradientBrush

6 element **LinearGradientBrush**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation


Opacity	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_ClrIntMode</a>		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	<a href="#">ST_Spread Method</a>		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	<a href="#">ST_Mapping Mode</a>	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	<a href="#">ST_RscRef Matrix</a>				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	<a href="#">ST_Point</a>	required			Specifies the starting point of the linear gradient.

	EndPoint	<u>ST_Point</u>	required		<p>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.</p>
annotation	Fills a region with a linear gradient.				

1 For more information, see §13.5 and §15.

## 2 19.28 LinearGradientBrush.GradientStops

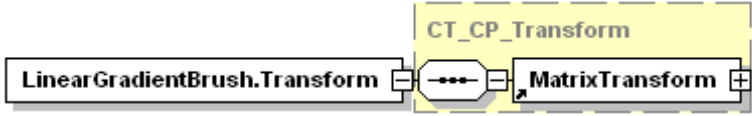
3 element **LinearGradientBrush.GradientStops**

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	<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>

7 For more information, see §14.4.8.

1 **19.30 LinkTarget**

2 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.					

3 For more information, see §10.2.3.

4 **19.31 ListItemStructure**

5 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.					

6 For more information, see §16.1.2.11.

1 **19.32 ListStructure**

2 element **ListStructure**

diagram	
annotation	Contains a collection of items that are group together in a list.

3 For more information, see §16.1.2.10.

4 **19.33 MatrixTransform**

5 element **MatrixTransform**

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>Matrix</td> <td><u>ST_Matrix</u></td> <td>required</td> <td></td> <td></td> <td>Specifies the matrix structure that defines the transformation.</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 [M7.11].</td> </tr> </tbody> </table>	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].
	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.																		

6 For more information, see §14.4.1.

7 **19.34 NamedElement**

8 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.					

1 For more information, see §16.1.2.13.

## 2 19.35 OutlineEntry

3 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.					

1 For more information, see §16.1.1.4.

## 2 19.36 PageContent

3 element **PageContent**

diagram						
attributes	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 OpenXPS Document [M3.5].
	Width	ST_GEOne				The width of the page contained in the page content.
	Height	ST_GEOne				The height of the page contained in the page content.
annotation	Defines a reference from a fixed document to a part that contains a <FixedPage> element.					

4 For more information, see §10.2.1.

## 5 19.37 PageContent.LinkTargets

6 element **PageContent.LinkTargets**

diagram	
annotation	<p>Contains a collection of &lt;LinkTarget&gt; elements, each of which is addressable via hyperlink.</p>

1 For more information, see §10.2.2.

## 2 **19.38 ParagraphStructure**

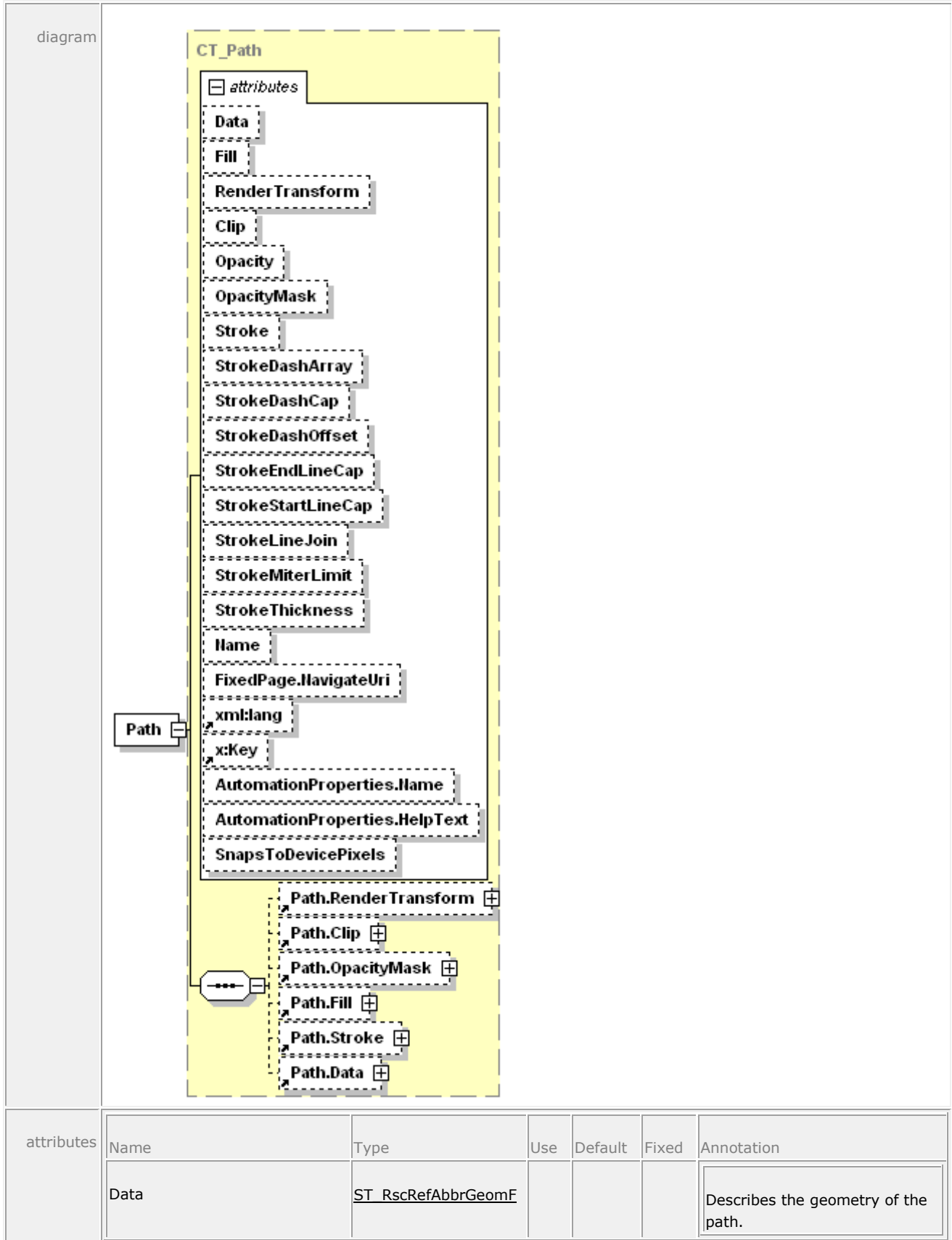
3 element **ParagraphStructure**

diagram	
annotation	<p>Contains the named elements that constitute a single paragraph.</p>

4 For more information, see §16.1.2.5.

## 5 **19.39 Path**

6 element **Path**



Fill	<a href="#">ST_RscRefColor</a>				Describes the brush used to paint the geometry specified by the Data property of the path.
RenderTransform	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_RscRefAbbrGeomF</a>				Limits the rendered region of the element.
Opacity	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRef</a>				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	<a href="#">ST_RscRefColor</a>				Specifies the brush used to draw the stroke.
StrokeDashArray	<a href="#">ST_EvenArrayPos</a>				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	<a href="#">ST_DashCap</a>		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 §14.3.2.

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**

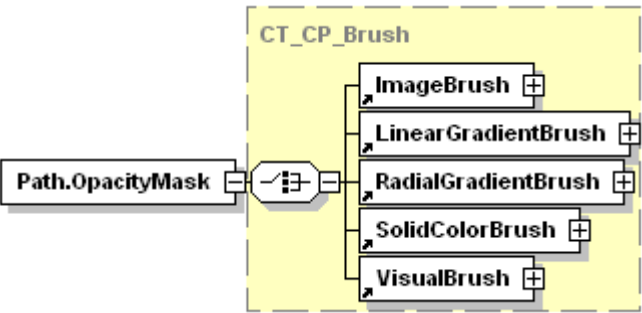
8 element **Path.Fill**

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**

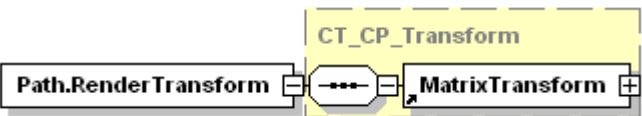
11 element **Path.OpacityMask**

<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

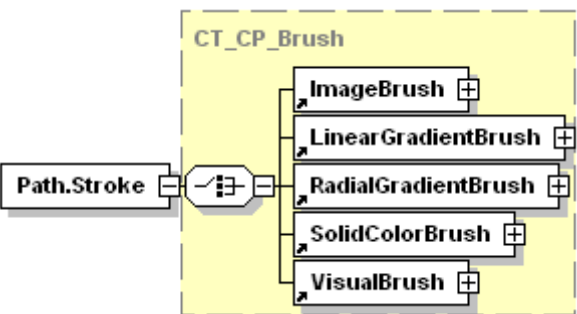
3 element **Path.RenderTransform**

<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 &lt;Path.Data&gt; property element.</p>

4 For more information, see §14.4.3.

## 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>

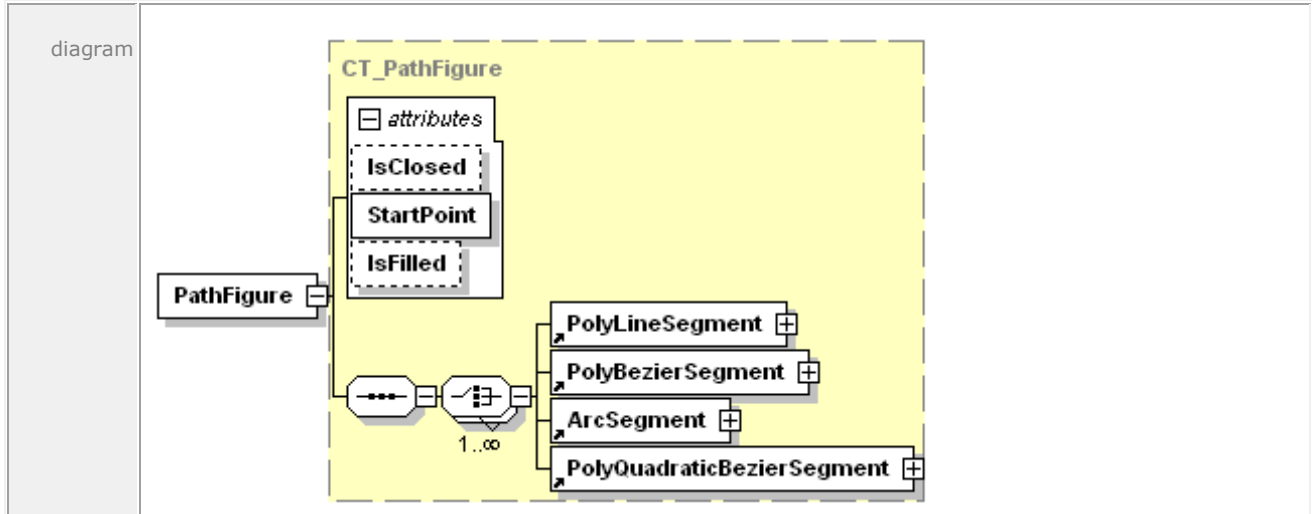
7



1 For more information, see §11.1.3.

2 **19.46 PathFigure**

3 element **PathFigure**



attributes	Name	Type	Use	Default	Fixed	Annotation
	IsClosed	<a href="#">ST_Boolean</a>		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	<a href="#">ST_Point</a>	required			Specifies the starting point for the first segment of the path figure.
	IsFilled	<a href="#">ST_Boolean</a>		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 &lt;PathFigure&gt; 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	Name	Type	Use	Default	Fixed	Annotation
	Points	<a href="#">ST_PointsM3</a>	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	<a href="#">ST_Boolean</a>		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	Name	Type	Use	Default	Fixed	Annotation
	Points	<a href="#">ST_Points</a>	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	<a href="#">ST_Boolean</a>		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.				

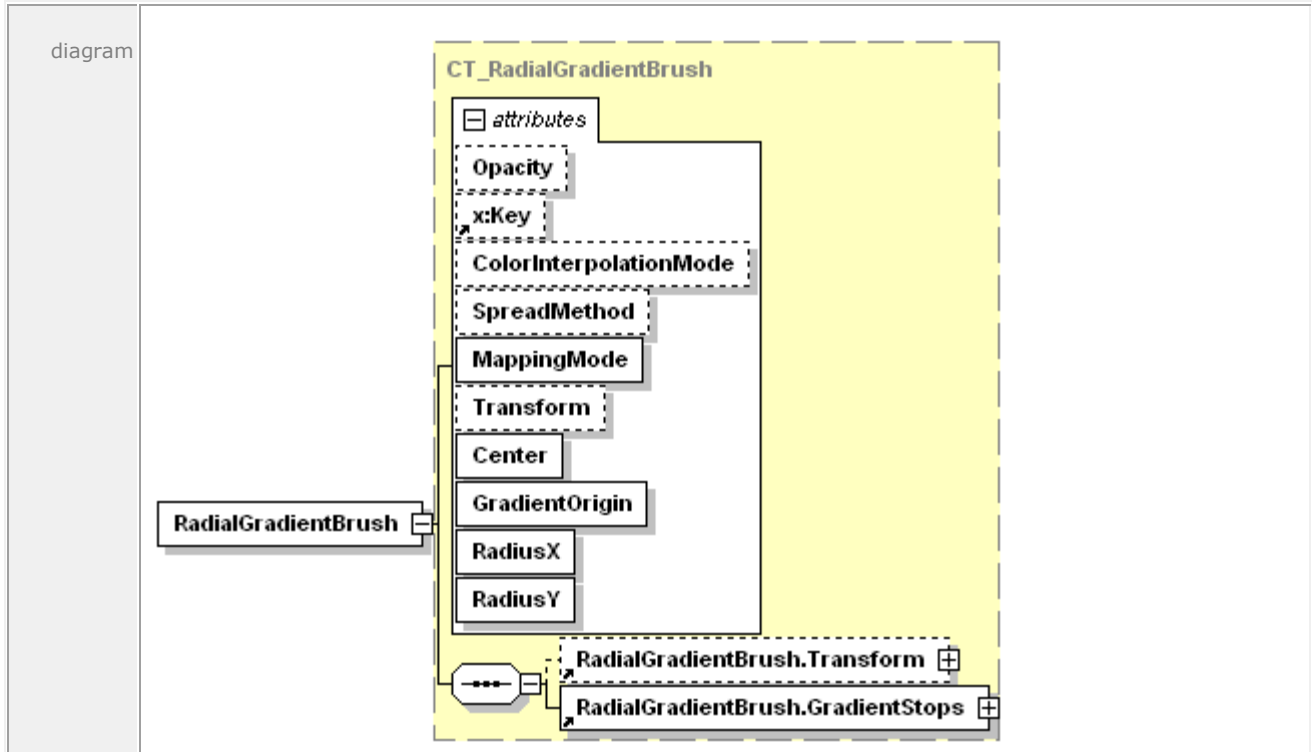
## 2 19.51 PolyQuadraticBezierSegment

3 element **PolyQuadraticBezierSegment**

diagram						
attributes	Name	Type	Use	Default	Fixed	Annotation
	Points	<a href="#">ST_PointsM2</a>	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	<a href="#">ST_Boolean</a>		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**




Name	Type	Use	Default	Fixed	Annotation
Opacity	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_ClrIntMode</a>		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	<a href="#">ST_Spread Method</a>		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	<a href="#">ST_MappingMode</a>	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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_Point</a>	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	<a href="#">ST_Point</a>	required			Specifies the origin point of the radial gradient.
	RadiusX	<a href="#">ST_GEZero</a>	required			Specifies the radius in the x dimension of the ellipse which defines the radial gradient.
	RadiusY	<a href="#">ST_GEZero</a>	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**

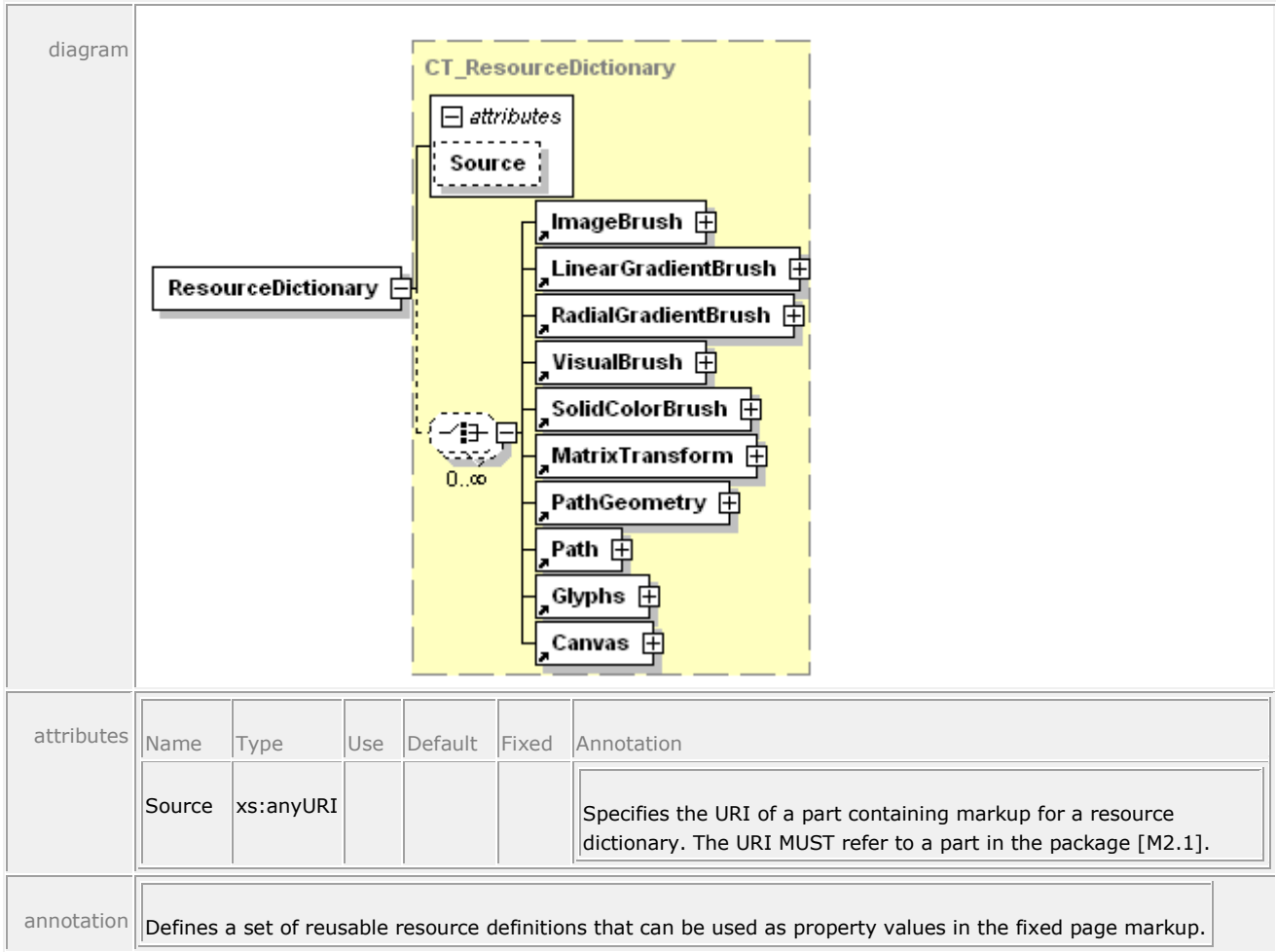
5 element **RadialGradientBrush.Transform**

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**

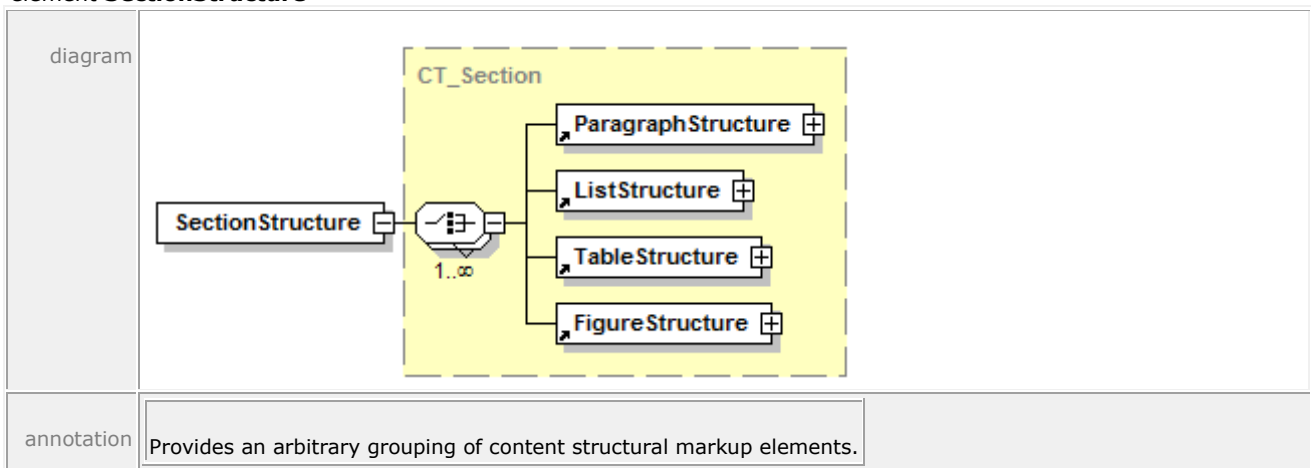
8 element **ResourceDictionary**



1 For more information, see §14.2.3.

## 2 19.56 SectionStructure

3 element **SectionStructure**



4 For more information, see §16.1.2.4.



1 **19.57 SignBy**

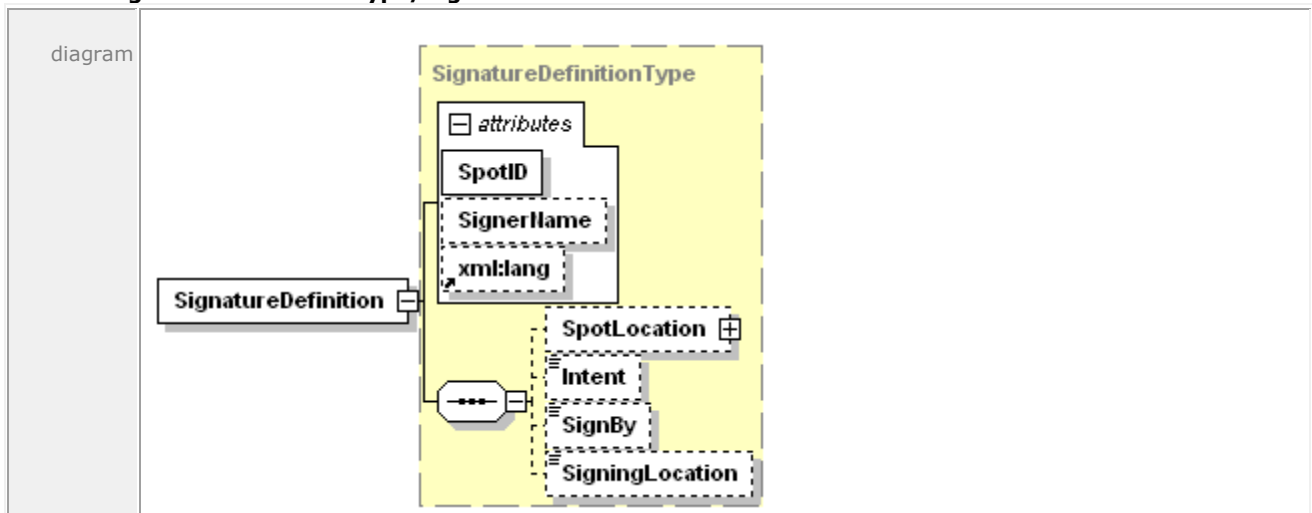
2 element **SignatureDefinitionType/SignBy**

diagram	
annotation	The date and time by which the requested party is to sign the OpenXPS 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			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 OpenXPS Document, or the name of the individual who has signed the OpenXPS 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.
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	<pre> classDiagram     class SpotLocationType {         +attributes         +PageURI         +StartX         +StartY     }     class SpotLocation     SpotLocationType &lt; -- SpotLocation             </pre>					
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
	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 OpenXPS Document.					

1 For more information, see §16.1.1.5.

## 2 19.64 StoryBreak

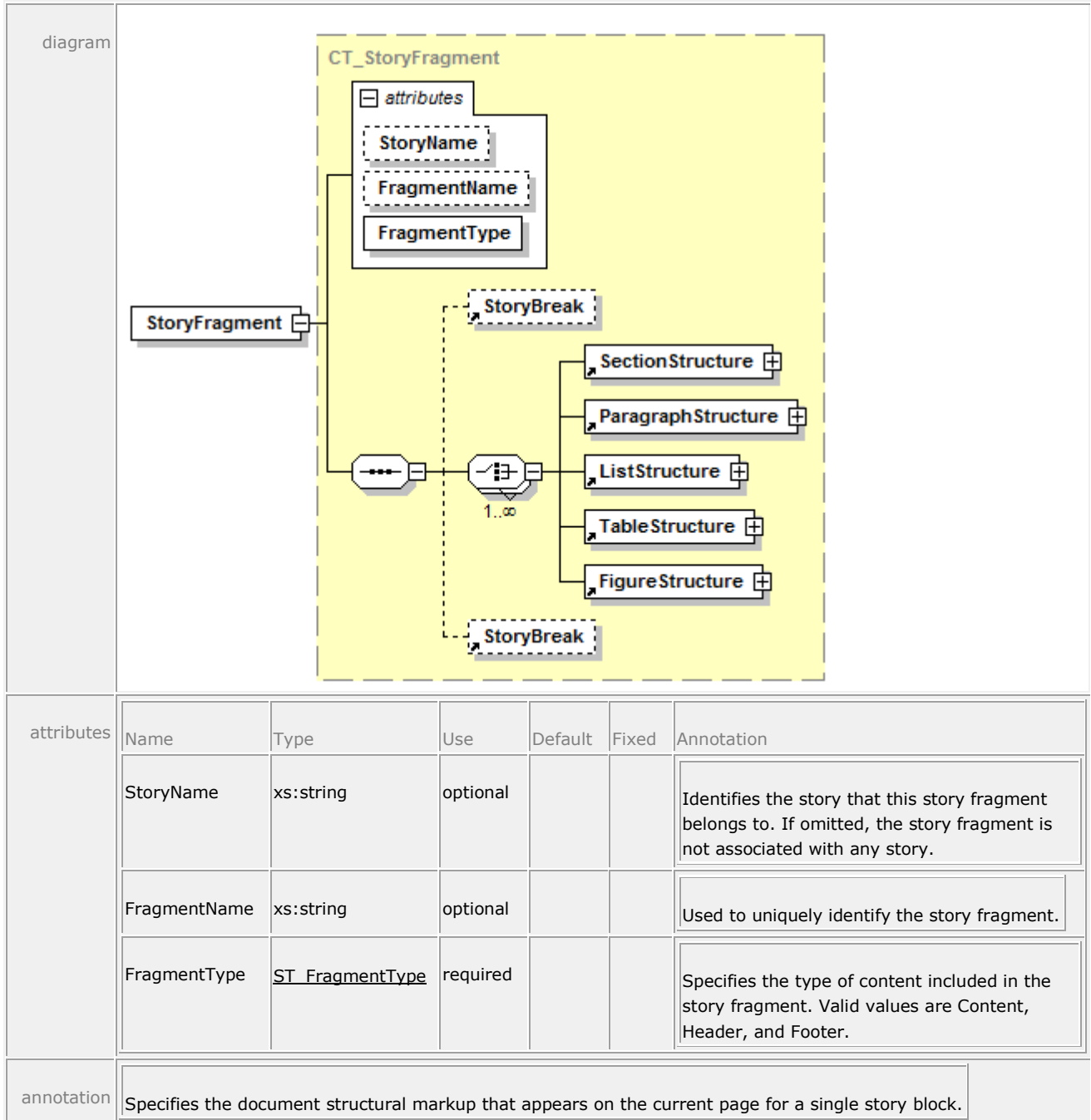
3 element **StoryBreak**

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>.

4 For more information, see §16.1.2.3.

## 5 19.65 StoryFragment

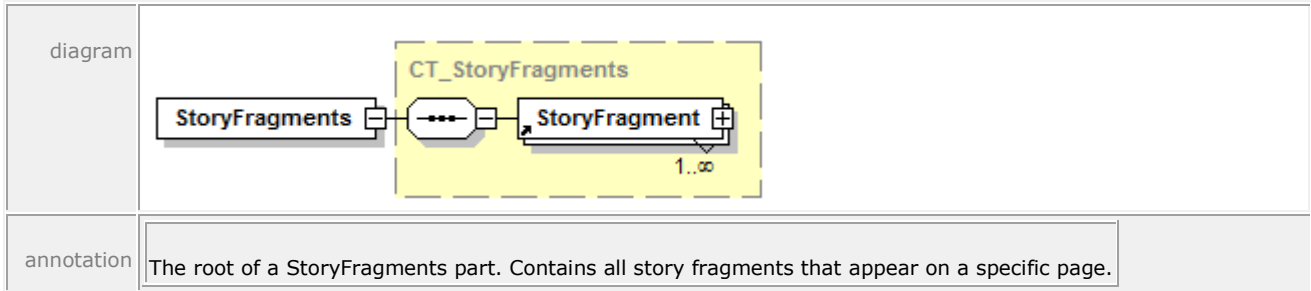
6 element **StoryFragment**



1 For more information, see §16.1.2.2.

## 2 19.66 StoryFragments

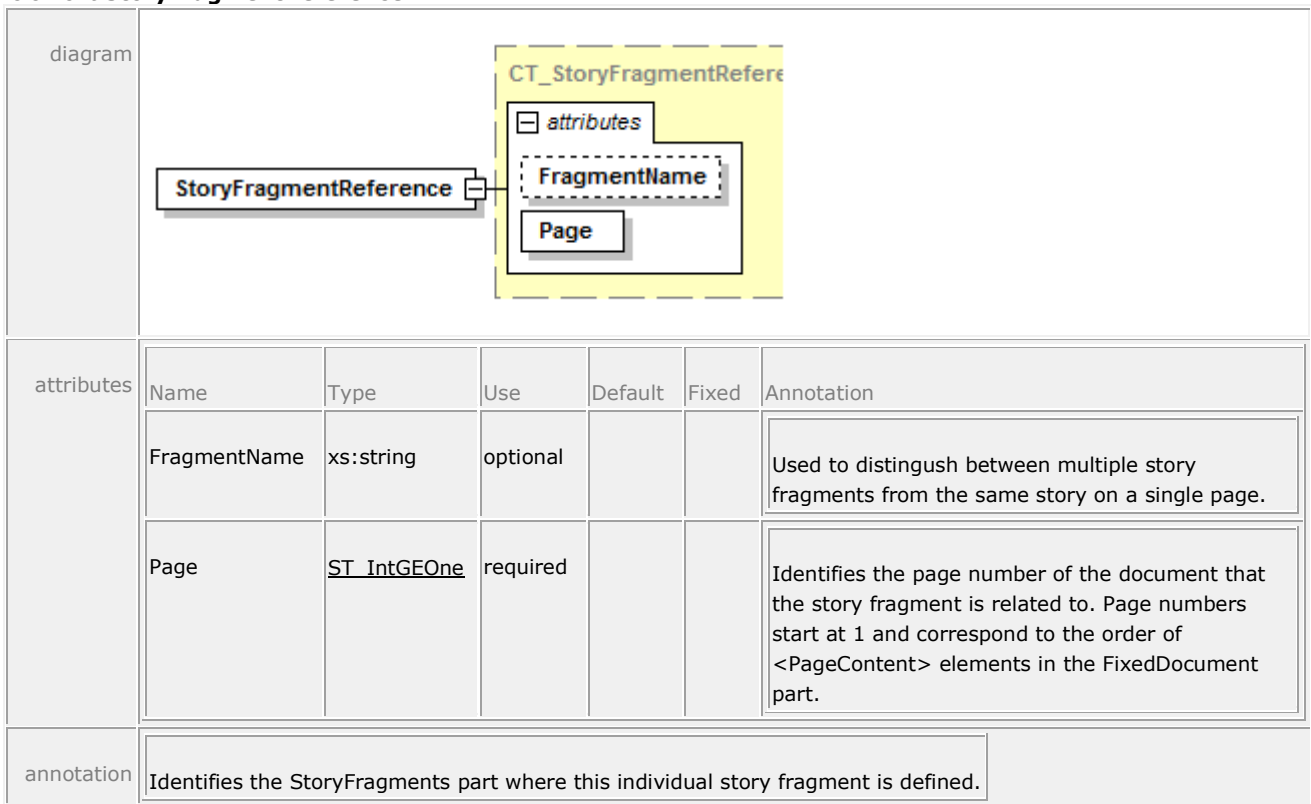
3 element **StoryFragments**



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>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>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**

diagram	<p>The diagram shows a class <b>TableRowStructure</b> on the left and a class <b>TableCellStructure</b> on the right. A composition relationship is shown between them, with a multiplicity of <b>1..∞</b> at the <b>TableCellStructure</b> end. The entire relationship is enclosed in a dashed yellow box labeled <b>CT_TableRow</b>.</p>
annotation	<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**

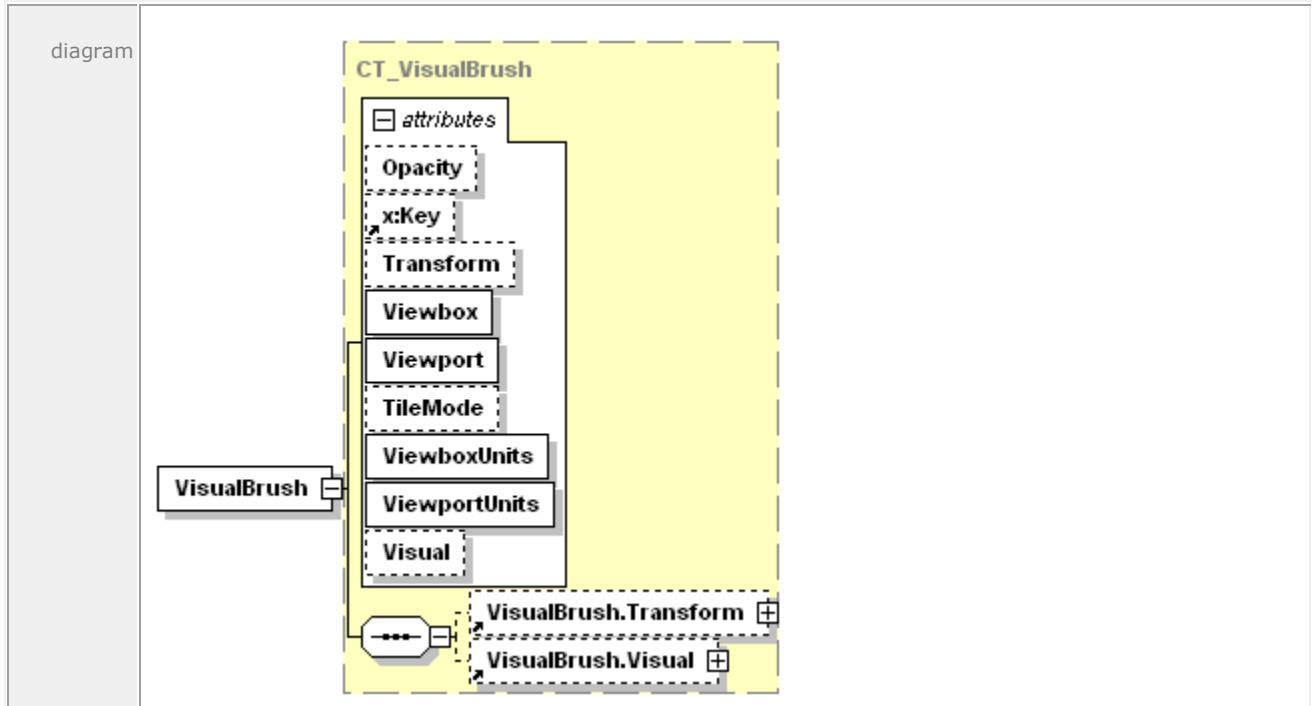
diagram	<p>The diagram shows a class <b>TableStructure</b> on the left and a class <b>TableRowGroupStructure</b> on the right. A composition relationship is shown between them, with a multiplicity of <b>1..∞</b> at the <b>TableRowGroupStructure</b> end. The entire relationship is enclosed in a dashed yellow box labeled <b>CT_Table</b>.</p>
annotation	<p>Contains a complete definition of a table in the OpenXPS 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	<a href="#">ST_ZeroOne</a>		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	<a href="#">ST_RscRefMatrix</a>				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	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_ViewBox</a>	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	<a href="#">ST_TileMode</a>		None		Specifies how contents will be tiled in the filled region. Valid values are None, Tile, FlipX, FlipY, and FlipXY.
ViewboxUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewbox coordinates to the containing coordinate space.
ViewportUnits	<a href="#">ST_ViewUnits</a>	required		Absolute	Specifies the relationship of the viewport coordinates to the containing coordinate space.
Visual	<a href="#">ST_RscRef</a>				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

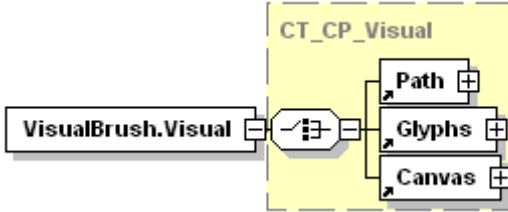
3 element **VisualBrush.Transform**

diagram	<pre> classDiagram     class VisualBrushTransform[VisualBrush.Transform]     class MatrixTransform     VisualBrushTransform -- MatrixTransform     subgraph CT_CP_Transform         MatrixTransform     end         </pre>
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 §14.4.7.

2 **19.74 VisualBrush.Visual**

3 element **VisualBrush.Visual**

<p>diagram</p>	 <p>The diagram shows a class <b>VisualBrush.Visual</b> on the left. A line with an open arrowhead points from <b>VisualBrush.Visual</b> to a circle containing a brush icon. This circle is connected to a dashed yellow box labeled <b>CT_CP_Visual</b>. Inside this box, three elements are listed: <b>Path</b>, <b>Glyphs</b>, and <b>Canvas</b>, each with a small square icon and a plus sign to its right.</p>
<p>annotation</p>	<p>Specifies a &lt;Path&gt; element, &lt;Glyphs&gt; element, or &lt;Canvas&gt; element used to draw the brush's source contents.</p>

4

5

6



## A. Schemas – W3C XML

### A.1 Signature Definitions

The schema shown below is also provided in electronic form as a file named [OpenXPSSignatureDefinitions.xsd](#), which is contained in an accompanying zip archive named "[OpenXPS WC3 Schemas.zip](#)". If discrepancies exist between the representation as published below and the corresponding electronic version, the published version below is the definitive version.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
targetNamespace="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/
signature-definitions"
xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/signature-
definitions" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
blockDefault="#all">

  <xs:import namespace="http://www.w3.org/XML/1998/namespace" />

  <xs:complexType name="SignatureDefinitionsType">
    <xs:sequence>
      <xs:element name="SignatureDefinition" type="SignatureDefinitionType"
minOccurs="1" maxOccurs="unbounded">
        </xs:element>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="SpotLocationType">
      <xs:attribute name="PageURI" type="xs:anyURI" use="required">
        </xs:attribute>
      <xs:attribute name="StartX" type="xs:double" use="required">
        </xs:attribute>
      <xs:attribute name="StartY" type="xs:double" use="required">
        </xs:attribute>
    </xs:complexType>

    <xs:complexType name="SignatureDefinitionType">
      <xs:sequence>
        <xs:element name="SpotLocation" type="SpotLocationType" minOccurs="0">
          </xs:element>
        <xs:element name="Intent" type="xs:string" minOccurs="0">
          </xs:element>
        <xs:element name="SignBy" type="xs:dateTime" minOccurs="0">
          </xs:element>
        <xs:element name="SigningLocation" type="xs:string" minOccurs="0">
          </xs:element>
        </xs:sequence>
      <xs:attribute name="SpotID" type="xs:ID" use="required">
        </xs:attribute>
    </xs:complexType>
  </xs:schema>
```

```
41     <xs:attribute name="SignerName" type="xs:string">
42         </xs:attribute>
43     <xs:attribute ref="xml:lang">
44         </xs:attribute>
45 </xs:complexType>
46
47     <xs:element name="SignatureDefinitions" type="SignatureDefinitionsType">
48         </xs:element>
49 </xs:schema>
```

## 1 A.2 OpenXPS Document

2 The schema shown below is also provided in electronic form as a file named  
 3 [OpenXPSDocument.xsd](#), which is contained in an accompanying zip archive named "[OpenXPS](#)  
 4 [WC3 Schemas.zip](#)". If discrepancies exist between the representation as published below and  
 5 the corresponding electronic version, 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/2005/06schemas.openxps.org/oxps/v1.0"
10  xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
11  xmlns:mstns="http://tempuri.org/XMLSchema.xsd"
12  xmlns:xs="http://www.w3.org/2001/XMLSchema"
13  xmlns:x="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resource
14  dictionary-key" elementFormDefault="qualified" blockDefault="#all">
15
16   <xs:import namespace=
17     "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedictiona
18     ry-key" />
19   <xs:import namespace="http://www.w3.org/XML/1998/namespace" />
20
21   <!-- Names used for types and groups:
22
23     ST_*      simpleType
24     CT_*      complexType
25     G_*       group
26     AG_*      attributeGroup
27
28   -->
29
30   <!-- Individual real number patterns
31   All patterns using numbers now use <whitespace value="collapse">.
32   As a result, any whitespace in the pattern can be expressed as:
33   mandatory whitespace, one or more: " "
34   optional whitespace, zero or more: " ?"
35
36   For better readability, each pattern using numbers is also described in a comment
37   using
38   one of the following pattern designators.
39
40   The actual patterns are generated by replacement by the schema publication process.
41   -->
42   <!--DEFINE [pint]      "[1-9][0-9]*" -->
43   <!--DEFINE [uint]     "[0-9]+" -->
44   <!--DEFINE [dec]      "(\\-?(((0-9)+(\\. [0-9]+)?)| (\\. [0-9]+)))" -->
45   <!--DEFINE [rn]       "((\\-|\\+)?(((0-9)+(\\. [0-9]+)?)| (\\. [0-9]+)))(e|E)(\\-|\\+)?[0-
46   9]+)?" -->
47   <!--DEFINE [prn]     "(\\+?(((0-9)+(\\. [0-9]+)?)| (\\. [0-9]+)))(e|E)(\\-|\\+)?[0-
48   9]+)?" -->
49   <!--DEFINE [scs]     "( ?, ?)" -->

```

```

46
47
48     <!-- Complex Types -->
49     <xs:complexType name="CT_MatrixTransform">
50         <xs:attributeGroup ref="AG_MatrixTransform" />
51     </xs:complexType>
52
53     <xs:complexType name="CT_SolidColorBrush">
54         <xs:attributeGroup ref="AG_Brush" />
55         <xs:attributeGroup ref="AG_SolidColorBrush" />
56     </xs:complexType>
57
58     <xs:complexType name="CT_ImageBrush">
59         <xs:sequence>
60             <xs:element ref="ImageBrush.Transform" minOccurs="0" />
61         </xs:sequence>
62         <xs:attributeGroup ref="AG_Brush" />
63         <xs:attributeGroup ref="AG_TileBrush" />
64         <xs:attributeGroup ref="AG_ImageBrush" />
65     </xs:complexType>
66
67     <xs:complexType name="CT_VisualBrush">
68         <xs:sequence>
69             <xs:element ref="VisualBrush.Transform" minOccurs="0" />
70             <xs:element ref="VisualBrush.Visual" minOccurs="0" />
71         </xs:sequence>
72         <xs:attributeGroup ref="AG_Brush" />
73         <xs:attributeGroup ref="AG_TileBrush" />
74         <xs:attributeGroup ref="AG_VisualBrush" />
75     </xs:complexType>
76
77     <xs:complexType name="CT_LinearGradientBrush">
78         <xs:sequence>
79             <xs:element ref="LinearGradientBrush.Transform" minOccurs="0" />
80             <xs:element ref="LinearGradientBrush.GradientStops" />
81         </xs:sequence>
82         <xs:attributeGroup ref="AG_Brush" />
83         <xs:attributeGroup ref="AG_GradientBrush" />
84         <xs:attributeGroup ref="AG_LinearGradientBrush" />
85     </xs:complexType>
86
87     <xs:complexType name="CT_RadialGradientBrush">
88         <xs:sequence>
89             <xs:element ref="RadialGradientBrush.Transform" minOccurs="0" />
90             <xs:element ref="RadialGradientBrush.GradientStops" />
91         </xs:sequence>
92         <xs:attributeGroup ref="AG_Brush" />
93         <xs:attributeGroup ref="AG_GradientBrush" />
94         <xs:attributeGroup ref="AG_RadialGradientBrush" />
95     </xs:complexType>
96
97     <xs:complexType name="CT_GradientStop">
98         <xs:attributeGroup ref="AG_GradientStop" />
99     </xs:complexType>
100

```



```

101 <xs:complexType name="CT_PathGeometry">
102   <xs:sequence>
103     <xs:element ref="PathGeometry.Transform" minOccurs="0" />
104     <xs:element ref="PathFigure" minOccurs="0" maxOccurs="unbounded" />
105   </xs:sequence>
106   <xs:attributeGroup ref="AG_PathGeometry" />
107 </xs:complexType>
108
109 <xs:complexType name="CT_Glyphs">
110   <xs:sequence>
111     <xs:element ref="Glyphs.RenderTransform" minOccurs="0" />
112     <xs:element ref="Glyphs.Clip" minOccurs="0" />
113     <xs:element ref="Glyphs.OpacityMask" minOccurs="0" />
114     <xs:element ref="Glyphs.Fill" minOccurs="0" />
115   </xs:sequence>
116   <xs:attributeGroup ref="AG_Glyphs" />
117 </xs:complexType>
118
119 <xs:complexType name="CT_Path">
120   <xs:sequence>
121     <xs:element ref="Path.RenderTransform" minOccurs="0" />
122     <xs:element ref="Path.Clip" minOccurs="0" />
123     <xs:element ref="Path.OpacityMask" minOccurs="0" />
124     <xs:element ref="Path.Fill" minOccurs="0" />
125     <xs:element ref="Path.Stroke" minOccurs="0" />
126     <xs:element ref="Path.Data" minOccurs="0" />
127   </xs:sequence>
128   <xs:attributeGroup ref="AG_Path" />
129   <xs:attributeGroup ref="AG_AutomationProvider" />
130   <xs:attributeGroup ref="AG_SnapsToDevicePixels" />
131 </xs:complexType>
132
133 <xs:complexType name="CT_PathFigure">
134   <xs:sequence>
135     <xs:choice maxOccurs="unbounded">
136       <xs:element ref="PolyLineSegment" />
137       <xs:element ref="PolyBezierSegment" />
138       <xs:element ref="ArcSegment" />
139       <xs:element ref="PolyQuadraticBezierSegment" />
140     </xs:choice>
141   </xs:sequence>
142   <xs:attributeGroup ref="AG_PathFigure" />
143 </xs:complexType>
144
145 <xs:complexType name="CT_ArcSegment">
146   <xs:attributeGroup ref="AG_ArcSegment" />
147 </xs:complexType>
148
149 <xs:complexType name="CT_PolyQuadraticBezierSegment">
150   <xs:attributeGroup ref="AG_PolyQuadraticBezierSegment" />
151 </xs:complexType>
152
153 <xs:complexType name="CT_PolyLineSegment">
154   <xs:attributeGroup ref="AG_PolyLineSegment" />
155 </xs:complexType>

```

```

156
157 <xs:complexType name="CT_PolyBezierSegment">
158   <xs:attributeGroup ref="AG_PolyBezierSegment" />
159 </xs:complexType>
160
161 <xs:complexType name="CT_Canvas">
162   <xs:sequence>
163     <xs:element ref="Canvas.Resources" minOccurs="0" />
164     <xs:element ref="Canvas.RenderTransform" minOccurs="0" />
165     <xs:element ref="Canvas.Clip" minOccurs="0" />
166     <xs:element ref="Canvas.OpacityMask" minOccurs="0" />
167     <xs:choice minOccurs="0" maxOccurs="unbounded">
168       <xs:element ref="Path" />
169       <xs:element ref="Glyphs" />
170       <xs:element ref="Canvas" />
171     </xs:choice>
172   </xs:sequence>
173   <xs:attributeGroup ref="AG_Canvas" />
174   <xs:attributeGroup ref="AG_AutomationProvider" />
175 </xs:complexType>
176
177 <xs:complexType name="CT_ResourceDictionary">
178   <xs:choice minOccurs="0" maxOccurs="unbounded">
179     <xs:element ref="ImageBrush" />
180     <xs:element ref="LinearGradientBrush" />
181     <xs:element ref="RadialGradientBrush" />
182     <xs:element ref="VisualBrush" />
183     <xs:element ref="SolidColorBrush" />
184     <xs:element ref="MatrixTransform" />
185     <xs:element ref="PathGeometry" />
186     <xs:element ref="Path" />
187     <xs:element ref="Glyphs" />
188     <xs:element ref="Canvas" />
189   </xs:choice>
190   <xs:attributeGroup ref="AG_ResourceDictionary" />
191 </xs:complexType>
192
193 <xs:complexType name="CT_FixedPage">
194   <xs:sequence>
195     <xs:element ref="FixedPage.Resources" minOccurs="0" />
196     <xs:choice minOccurs="0" maxOccurs="unbounded">
197       <xs:element ref="Path" />
198       <xs:element ref="Glyphs" />
199       <xs:element ref="Canvas" />
200     </xs:choice>
201   </xs:sequence>
202   <xs:attributeGroup ref="AG_FixedPage" />
203 </xs:complexType>
204
205 <xs:complexType name="CT_FixedDocument">
206   <xs:sequence>
207     <xs:element ref="PageContent" maxOccurs="unbounded" />
208   </xs:sequence>
209 </xs:complexType>
210

```

```
211 <xs:complexType name="CT_PageContent">
212   <xs:sequence>
213     <xs:element ref="PageContent.LinkTargets" minOccurs="0" />
214   </xs:sequence>
215   <xs:attributeGroup ref="AG_PageContent" />
216 </xs:complexType>
217
218 <xs:complexType name="CT_FixedDocumentSequence">
219   <xs:sequence>
220     <xs:element ref="DocumentReference" maxOccurs="unbounded" />
221   </xs:sequence>
222 </xs:complexType>
223
224 <xs:complexType name="CT_DocumentReference">
225   <xs:attributeGroup ref="AG_DocumentReference" />
226 </xs:complexType>
227
228 <xs:complexType name="CT_LinkTarget">
229   <xs:attributeGroup ref="AG_LinkTarget" />
230 </xs:complexType>
231
232 <xs:complexType name="CT_CP_LinkTargets">
233   <xs:sequence>
234     <xs:element ref="LinkTarget" maxOccurs="unbounded" />
235   </xs:sequence>
236 </xs:complexType>
237
238 <xs:complexType name="CT_CP_Transform">
239   <xs:sequence>
240     <xs:element ref="MatrixTransform" />
241   </xs:sequence>
242 </xs:complexType>
243
244 <xs:complexType name="CT_CP_Visual">
245   <xs:choice>
246     <xs:element ref="Path" />
247     <xs:element ref="Glyphs" />
248     <xs:element ref="Canvas" />
249   </xs:choice>
250 </xs:complexType>
251
252 <xs:complexType name="CT_CP_GradientStops">
253   <xs:sequence>
254     <xs:element ref="GradientStop" minOccurs="2" maxOccurs="unbounded" />
255   </xs:sequence>
256 </xs:complexType>
257
258 <xs:complexType name="CT_CP_Geometry">
259   <xs:sequence>
260     <xs:element ref="PathGeometry" />
261   </xs:sequence>
262 </xs:complexType>
263
264 <xs:complexType name="CT_CP_Brush">
265   <xs:choice>
```

```

266         <xs:element ref="ImageBrush" />
267         <xs:element ref="LinearGradientBrush" />
268         <xs:element ref="RadialGradientBrush" />
269         <xs:element ref="SolidColorBrush" />
270         <xs:element ref="VisualBrush" />
271     </xs:choice>
272 </xs:complexType>
273
274 <xs:complexType name="CT_CP_Resources">
275     <xs:sequence minOccurs="0">
276         <xs:element ref="ResourceDictionary" />
277     </xs:sequence>
278 </xs:complexType>
279
280 <!-- Root elements -->
281 <xs:element name="MatrixTransform" type="CT_MatrixTransform">
282     </xs:element>
283
284 <xs:element name="SolidColorBrush" type="CT_SolidColorBrush">
285     </xs:element>
286
287 <xs:element name="ImageBrush" type="CT_ImageBrush">
288     </xs:element>
289
290 <xs:element name="VisualBrush" type="CT_VisualBrush">
291     </xs:element>
292
293 <xs:element name="LinearGradientBrush" type="CT_LinearGradientBrush">
294     </xs:element>
295
296 <xs:element name="RadialGradientBrush" type="CT_RadialGradientBrush">
297     </xs:element>
298
299 <xs:element name="Glyphs" type="CT_Glyphs">
300     </xs:element>
301
302 <xs:element name="Path" type="CT_Path">
303     </xs:element>
304
305 <xs:element name="Canvas" type="CT_Canvas">
306     </xs:element>
307
308 <xs:element name="GradientStop" type="CT_GradientStop">
309     </xs:element>
310
311 <xs:element name="ResourceDictionary" type="CT_ResourceDictionary">
312     </xs:element>
313
314 <xs:element name="PathGeometry" type="CT_PathGeometry">
315     </xs:element>
316
317 <xs:element name="PathFigure" type="CT_PathFigure">
318     </xs:element>
319
320 <xs:element name="PolyLineSegment" type="CT_PolyLineSegment">

```

```
321     </xs:element>
322
323     <xs:element name="ArcSegment" type="CT_ArcSegment">
324     </xs:element>
325
326     <xs:element name="PolyBezierSegment" type="CT_PolyBezierSegment">
327     </xs:element>
328
329     <xs:element name="PolyQuadraticBezierSegment" type="CT_PolyQuadraticBezierSegment">
330     </xs:element>
331
332     <xs:element name="FixedPage" type="CT_FixedPage">
333     </xs:element>
334
335     <xs:element name="FixedDocument" type="CT_FixedDocument">
336     </xs:element>
337
338     <xs:element name="PageContent" type="CT_PageContent">
339     </xs:element>
340
341     <xs:element name="FixedDocumentSequence" type="CT_FixedDocumentSequence">
342     </xs:element>
343
344     <xs:element name="DocumentReference" type="CT_DocumentReference">
345     </xs:element>
346
347     <xs:element name="LinkTarget" type="CT_LinkTarget">
348     </xs:element>
349
350     <xs:element name="PageContent.LinkTargets" type="CT_CP_LinkTargets">
351     </xs:element>
352
353     <xs:element name="ImageBrush.Transform" type="CT_CP_Transform">
354     </xs:element>
355
356     <xs:element name="VisualBrush.Transform" type="CT_CP_Transform">
357     </xs:element>
358
359     <xs:element name="LinearGradientBrush.Transform" type="CT_CP_Transform">
360     </xs:element>
361
362     <xs:element name="RadialGradientBrush.Transform" type="CT_CP_Transform">
363     </xs:element>
364
365     <xs:element name="PathGeometry.Transform" type="CT_CP_Transform">
366     </xs:element>
367
368     <xs:element name="Glyphs.RenderTransform" type="CT_CP_Transform">
369     </xs:element>
370
371     <xs:element name="Path.RenderTransform" type="CT_CP_Transform">
372     </xs:element>
373
374     <xs:element name="Canvas.RenderTransform" type="CT_CP_Transform">
375     </xs:element>
```

```

376
377 <xs:element name="VisualBrush.Visual" type="CT_CP_Visual">
378     </xs:element>
379
380 <xs:element name="LinearGradientBrush.GradientStops" type="CT_CP_GradientStops">
381     </xs:element>
382
383 <xs:element name="RadialGradientBrush.GradientStops" type="CT_CP_GradientStops">
384     </xs:element>
385
386 <xs:element name="Glyphs.Clip" type="CT_CP_Geometry">
387     </xs:element>
388
389 <xs:element name="Path.Clip" type="CT_CP_Geometry">
390     </xs:element>
391
392 <xs:element name="Canvas.Clip" type="CT_CP_Geometry">
393     </xs:element>
394
395 <xs:element name="Glyphs.OpacityMask" type="CT_CP_Brush">
396     </xs:element>
397
398 <xs:element name="Path.OpacityMask" type="CT_CP_Brush">
399     </xs:element>
400
401 <xs:element name="Canvas.OpacityMask" type="CT_CP_Brush">
402     </xs:element>
403
404 <xs:element name="Glyphs.Fill" type="CT_CP_Brush">
405     </xs:element>
406
407 <xs:element name="Path.Fill" type="CT_CP_Brush">
408     </xs:element>
409
410 <xs:element name="Path.Data" type="CT_CP_Geometry">
411     </xs:element>
412
413 <xs:element name="Path.Stroke" type="CT_CP_Brush">
414     </xs:element>
415
416 <xs:element name="Canvas.Resources" type="CT_CP_Resources">
417     </xs:element>
418
419 <xs:element name="FixedPage.Resources" type="CT_CP_Resources">
420     </xs:element>
421
422 <xs:attributeGroup name="AG_GradientStop">
423     <xs:attribute name="Color" type="ST_Color" use="required">
424         </xs:attribute>
425     <xs:attribute name="Offset" type="ST_Double" use="required">
426         </xs:attribute>
427 </xs:attributeGroup>
428
429 <xs:attributeGroup name="AG_Brush">
430     <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">

```

```

431         </xs:attribute>
432         <xs:attribute ref="x:Key" />
433     </xs:attributeGroup>
434
435     <xs:attributeGroup name="AG_TileBrush">
436         <xs:attribute name="Transform" type="ST_RscRefMatrix">
437             </xs:attribute>
438         <xs:attribute name="Viewbox" type="ST_ViewBox" use="required">
439             </xs:attribute>
440         <xs:attribute name="Viewport" type="ST_ViewBox" use="required">
441             </xs:attribute>
442         <xs:attribute name="TileMode" type="ST_TileMode" default="None">
443             </xs:attribute>
444         <xs:attribute name="ViewboxUnits" type="ST_ViewUnits" use="required"
445 fixed="Absolute">
446             </xs:attribute>
447         <xs:attribute name="ViewportUnits" type="ST_ViewUnits" use="required"
448 fixed="Absolute">
449             </xs:attribute>
450     </xs:attributeGroup>
451
452     <xs:attributeGroup name="AG_VisualBrush">
453         <xs:attribute name="Visual" type="ST_RscRef">
454             </xs:attribute>
455     </xs:attributeGroup>
456
457     <xs:attributeGroup name="AG_GradientBrush">
458         <xs:attribute name="ColorInterpolationMode" type="ST_ClrIntMode"
459 default="SRgbLinearInterpolation">
460             </xs:attribute>
461         <xs:attribute name="SpreadMethod" type="ST_SpreadMethod" default="Pad">
462             </xs:attribute>
463         <xs:attribute name="MappingMode" type="ST_MappingMode" use="required"
464 fixed="Absolute">
465             </xs:attribute>
466     </xs:attributeGroup>
467
468     <xs:attributeGroup name="AG_SolidColorBrush">
469         <xs:attribute name="Color" type="ST_Color" use="required">
470             </xs:attribute>
471     </xs:attributeGroup>
472
473     <xs:attributeGroup name="AG_ImageBrush">
474         <xs:attribute name="ImageSource" type="ST_UriCtxBmp" use="required">
475             </xs:attribute>
476     </xs:attributeGroup>
477
478     <xs:attributeGroup name="AG_LinearGradientBrush">
479         <xs:attribute name="Transform" type="ST_RscRefMatrix">
480             </xs:attribute>
481         <xs:attribute name="StartPoint" type="ST_Point" use="required">
482             </xs:attribute>
483         <xs:attribute name="EndPoint" type="ST_Point" use="required">
484             </xs:attribute>
485     </xs:attributeGroup>

```

```

486
487 <xs:attributeGroup name="AG_RadialGradientBrush">
488   <xs:attribute name="Transform" type="ST_RscRefMatrix">
489     </xs:attribute>
490   <xs:attribute name="Center" type="ST_Point" use="required">
491     </xs:attribute>
492   <xs:attribute name="GradientOrigin" type="ST_Point" use="required">
493     </xs:attribute>
494   <xs:attribute name="RadiusX" type="ST_GEZero" use="required">
495     </xs:attribute>
496   <xs:attribute name="RadiusY" type="ST_GEZero" use="required">
497     </xs:attribute>
498 </xs:attributeGroup>
499
500 <xs:attributeGroup name="AG_PathGeometry">
501   <xs:attribute name="Figures" type="ST_AbbrGeom">
502     </xs:attribute>
503   <xs:attribute name="FillRule" type="ST_FillRule" default="EvenOdd">
504     </xs:attribute>
505   <xs:attribute name="Transform" type="ST_RscRefMatrix">
506     </xs:attribute>
507   <xs:attribute ref="x:Key" />
508 </xs:attributeGroup>
509
510 <xs:attributeGroup name="AG_ResourceDictionary">
511   <xs:attribute name="Source" type="xs:anyURI">
512     </xs:attribute>
513 </xs:attributeGroup>
514
515 <xs:attributeGroup name="AG_PolyLineSegment">
516   <xs:attribute name="Points" type="ST_Points" use="required">
517     </xs:attribute>
518   <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
519     </xs:attribute>
520 </xs:attributeGroup>
521
522 <xs:attributeGroup name="AG_ArcSegment">
523   <xs:attribute name="Point" type="ST_Point" use="required">
524     </xs:attribute>
525   <xs:attribute name="Size" type="ST_PointGE0" use="required">
526     </xs:attribute>
527   <xs:attribute name="RotationAngle" type="ST_Double" use="required">
528     </xs:attribute>
529   <xs:attribute name="IsLargeArc" type="ST_Boolean" use="required">
530     </xs:attribute>
531   <xs:attribute name="SweepDirection" type="ST_SweepDirection" use="required">
532     </xs:attribute>
533   <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
534     </xs:attribute>
535 </xs:attributeGroup>
536
537 <xs:attributeGroup name="AG_PolyBezierSegment">
538   <xs:attribute name="Points" type="ST_PointsM3" use="required">
539     </xs:attribute>
540   <xs:attribute name="IsStroked" type="ST_Boolean" default="true">

```



```

541         </xs:attribute>
542     </xs:attributeGroup>
543
544     <xs:attributeGroup name="AG_PolyQuadraticBezierSegment">
545         <xs:attribute name="Points" type="ST_PointsM2" use="required">
546             </xs:attribute>
547         <xs:attribute name="IsStroked" type="ST_Boolean" default="true">
548             </xs:attribute>
549     </xs:attributeGroup>
550
551     <xs:attributeGroup name="AG_Glyphs">
552         <xs:attribute name="BidiLevel" default="0">
553             <xs:simpleType>
554                 <xs:restriction base="xs:integer">
555                     <xs:minInclusive value="0" />
556                     <xs:maxInclusive value="61" />
557                 </xs:restriction>
558             </xs:simpleType>
559         </xs:attribute>
560         <xs:attribute name="CaretStops" type="ST_CaretStops">
561             </xs:attribute>
562         <xs:attribute name="DeviceFontName" type="ST_UnicodeString">
563             </xs:attribute>
564         <xs:attribute name="Fill" type="ST_RscRefColor">
565             </xs:attribute>
566         <xs:attribute name="FontRenderingEmSize" type="ST_GEZero" use="required">
567             </xs:attribute>
568         <xs:attribute name="FontUri" type="xs:anyURI" use="required">
569             </xs:attribute>
570         <xs:attribute name="OriginX" type="ST_Double" use="required">
571             </xs:attribute>
572         <xs:attribute name="OriginY" type="ST_Double" use="required">
573             </xs:attribute>
574         <xs:attribute name="IsSideways" type="ST_Boolean" default="false">
575             </xs:attribute>
576         <xs:attribute name="Indices" type="ST_Indices">
577             </xs:attribute>
578         <xs:attribute name="UnicodeString" type="ST_UnicodeString">
579             </xs:attribute>
580         <xs:attribute name="StyleSimulations" type="ST_StyleSimulations"
581 default="None">
582             </xs:attribute>
583         <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">
584             </xs:attribute>
585         <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">
586             </xs:attribute>
587         <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
588             </xs:attribute>
589         <xs:attribute name="OpacityMask" type="ST_RscRef">
590             </xs:attribute>
591         <xs:attribute name="Name" type="ST_Name">
592             </xs:attribute>
593         <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
594             </xs:attribute>
595         <xs:attribute ref="xml:lang">

```

```

596         </xs:attribute>
597         <xs:attribute ref="x:Key" />
598     </xs:attributeGroup>
599
600     <xs:attributeGroup name="AG_Path">
601         <xs:attribute name="Data" type="ST_RscRefAbbrGeomF">
602             </xs:attribute>
603         <xs:attribute name="Fill" type="ST_RscRefColor">
604             </xs:attribute>
605         <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">
606             </xs:attribute>
607         <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">
608             </xs:attribute>
609         <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
610             </xs:attribute>
611         <xs:attribute name="OpacityMask" type="ST_RscRef">
612             </xs:attribute>
613         <xs:attribute name="Stroke" type="ST_RscRefColor">
614             </xs:attribute>
615         <xs:attribute name="StrokeDashArray" type="ST_EvenArrayPos">
616             </xs:attribute>
617         <xs:attribute name="StrokeDashCap" type="ST_DashCap" default="Flat">
618             </xs:attribute>
619         <xs:attribute name="StrokeDashOffset" type="ST_Double" default="0.0">
620             </xs:attribute>
621         <xs:attribute name="StrokeEndLineCap" type="ST_LineCap" default="Flat">
622             </xs:attribute>
623         <xs:attribute name="StrokeStartLineCap" type="ST_LineCap" default="Flat">
624             </xs:attribute>
625         <xs:attribute name="StrokeLineJoin" type="ST_LineJoin" default="Miter">
626             </xs:attribute>
627         <xs:attribute name="StrokeMiterLimit" type="ST_GEOne" default="10.0">
628             </xs:attribute>
629         <xs:attribute name="StrokeThickness" type="ST_GEZero" default="1.0">
630             </xs:attribute>
631         <xs:attribute name="Name" type="ST_Name">
632             </xs:attribute>
633         <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
634             </xs:attribute>
635         <xs:attribute ref="xml:lang">
636             </xs:attribute>
637         <xs:attribute ref="x:Key" />
638     </xs:attributeGroup>
639
640     <xs:attributeGroup name="AG_PathFigure">
641         <xs:attribute name="IsClosed" type="ST_Boolean" default="false">
642             </xs:attribute>
643         <xs:attribute name="StartPoint" type="ST_Point" use="required">
644             </xs:attribute>
645         <xs:attribute name="IsFilled" type="ST_Boolean" default="true">
646             </xs:attribute>
647     </xs:attributeGroup>
648
649     <xs:attributeGroup name="AG_Canvas">
650         <xs:attribute name="RenderTransform" type="ST_RscRefMatrix">

```

```

651         </xs:attribute>
652         <xs:attribute name="Clip" type="ST_RscRefAbbrGeomF">
653             </xs:attribute>
654         <xs:attribute name="Opacity" type="ST_ZeroOne" default="1.0">
655             </xs:attribute>
656         <xs:attribute name="OpacityMask" type="ST_RscRef">
657             </xs:attribute>
658         <xs:attribute name="Name" type="ST_Name">
659             </xs:attribute>
660         <xs:attribute name="RenderOptions.EdgeMode" type="ST_EdgeMode">
661             </xs:attribute>
662         <xs:attribute name="FixedPage.NavigateUri" type="xs:anyURI">
663             </xs:attribute>
664         <xs:attribute ref="xml:lang">
665             </xs:attribute>
666         <xs:attribute ref="x:Key" />
667     </xs:attributeGroup>
668
669     <xs:attributeGroup name="AG_PageContent">
670         <xs:attribute name="Source" type="xs:anyURI" use="required">
671             </xs:attribute>
672         <xs:attribute name="Width" type="ST_GEOne">
673             </xs:attribute>
674         <xs:attribute name="Height" type="ST_GEOne">
675             </xs:attribute>
676     </xs:attributeGroup>
677
678     <xs:attributeGroup name="AG_LinkTarget">
679         <xs:attribute name="Name" type="ST_NUName" use="required">
680             </xs:attribute>
681     </xs:attributeGroup>
682
683     <xs:attributeGroup name="AG_DocumentReference">
684         <xs:attribute name="Source" type="xs:anyURI" use="required">
685             </xs:attribute>
686     </xs:attributeGroup>
687
688     <xs:attributeGroup name="AG_MatrixTransform">
689         <xs:attribute name="Matrix" type="ST_Matrix" use="required">
690             </xs:attribute>
691         <xs:attribute ref="x:Key" />
692     </xs:attributeGroup>
693
694     <xs:attributeGroup name="AG_FixedPage">
695         <xs:attribute name="Width" type="ST_GEOne" use="required">
696             </xs:attribute>
697         <xs:attribute name="Height" type="ST_GEOne" use="required">
698             </xs:attribute>
699         <xs:attribute name="ContentBox" type="ST_ContentBox">
700             </xs:attribute>
701         <xs:attribute name="BleedBox" type="ST_BleedBox">
702             </xs:attribute>
703         <xs:attribute ref="xml:lang" use="required">
704             </xs:attribute>
705         <xs:attribute name="Name" type="ST_Name">

```

```

706         </xs:attribute>
707     </xs:attributeGroup>
708
709     <xs:attributeGroup name="AG_AutomationProvider">
710         <xs:attribute name="AutomationProperties.Name" type="xs:string">
711             </xs:attribute>
712         <xs:attribute name="AutomationProperties.HelpText" type="xs:string">
713             </xs:attribute>
714     </xs:attributeGroup>
715
716     <xs:attributeGroup name="AG_SnapsToDevicePixels">
717         <xs:attribute name="SnapsToDevicePixels" type="ST_Boolean">
718             </xs:attribute>
719     </xs:attributeGroup>
720
721     <!-- Simple data types -->
722     <!-- A unique Name (ID with pattern restriction according to OpenXPS spec) -->
723     <xs:simpleType name="ST_Name">
724         <xs:restriction base="xs:ID">
725             <xs:pattern
726 value="(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{N1}|_)(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{N1}|\p{Mn}
727 |\p{Mc}|\p{Nd}|_)*" />
728             </xs:restriction>
729         </xs:simpleType>
730
731     <!-- A non-unique Name (ID with pattern restriction according to OpenXPS spec) -->
732     <xs:simpleType name="ST_NUName">
733         <xs:restriction base="xs:string">
734             <xs:pattern
735 value="(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{N1}|_)(\p{Lu}|\p{Ll}|\p{Lt}|\p{Lo}|\p{N1}|\p{Mn}
736 |\p{Mc}|\p{Nd}|_)*" />
737             </xs:restriction>
738         </xs:simpleType>
739
740     <!-- Boolean with true and false only (no 0 or 1) -->
741     <xs:simpleType name="ST_Boolean">
742         <xs:restriction base="xs:boolean">
743             <xs:pattern value="true|false" />
744         </xs:restriction>
745     </xs:simpleType>
746
747     <!-- real number from 0.0 to 1.0 inclusive -->
748     <xs:simpleType name="ST_ZeroOne">
749         <xs:restriction base="ST_Double">
750             <xs:minInclusive value="0.0" />
751             <xs:maxInclusive value="1.0" />
752         </xs:restriction>
753     </xs:simpleType>
754
755     <!-- positive real number -->
756     <xs:simpleType name="ST_GEZero">
757         <xs:restriction base="ST_Double">
758             <xs:minInclusive value="0.0" />
759         </xs:restriction>
760     </xs:simpleType>

```

```

761
762 <!-- positive real number, equal or greater than one -->
763 <xs:simpleType name="ST_GEOne">
764   <xs:restriction base="ST_Double">
765     <xs:minInclusive value="1.0" />
766   </xs:restriction>
767 </xs:simpleType>
768
769 <!-- Double -->
770 <xs:simpleType name="ST_Double">
771   <xs:restriction base="xs:double">
772     <xs:whiteSpace value="collapse" />
773 <!--
774     <xs:pattern value="[rn]" />
775 -->
776     <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
777 |\+)?[\0-9]+))?" />
778   </xs:restriction>
779 </xs:simpleType>
780
781 <!-- Point: 2 numbers, separated by , and arbitrary whitespace -->
782 <xs:simpleType name="ST_Point">
783   <xs:restriction base="xs:string">
784     <xs:whiteSpace value="collapse" />
785 <!--
786     <xs:pattern value="[rn][scs][rn]" />
787 -->
788     <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
789 |\+)?[\0-9]+))( , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+))?"
790 />
791   </xs:restriction>
792 </xs:simpleType>
793
794 <!-- PointGE0: 2 non-negative numbers, separated by , and arbitrary whitespace -->
795 <xs:simpleType name="ST_PointGE0">
796   <xs:restriction base="xs:string">
797     <xs:whiteSpace value="collapse" />
798 <!--
799     <xs:pattern value="[prn][scs][prn]" />
800 -->
801     <xs:pattern value="(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-
802 9]+)?)( , ?)(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+))?" />
803   </xs:restriction>
804 </xs:simpleType>
805
806 <!-- Points: List of ST_Point, separated by arbitrary whitespace -->
807 <xs:simpleType name="ST_Points">
808   <xs:restriction base="xs:string">
809     <xs:whiteSpace value="collapse" />
810 <!--
811     <xs:pattern value="[rn][scs][rn]( [rn][scs][rn])*" />
812 -->
813     <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-
814 |\+)?[\0-9]+))( , ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+))("

```

```

815 ((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-
816 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?))*" />
817     </xs:restriction>
818     </xs:simpleType>
819
820     <!-- PointsM2: List of ST Point, separated by arbitrary whitespace with a multiple
821 of 2 point count -->
822     <xs:simpleType name="ST PointsM2">
823         <xs:restriction base="xs:string">
824             <xs:whiteSpace value="collapse" />
825         <!--
826             <xs:pattern value="[rn][scs][rn] [rn][scs][rn](( [rn][scs][rn]){2})*"/>
827         -->
828         <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-
829 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-
830 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-
831 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)(( (\-|\+)?(([\0-9]+(\.[\0-
832 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-9]+(\.[\0-
833 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)){2})*" />
834         </xs:restriction>
835     </xs:simpleType>
836
837     <!-- PointsM3: List of ST Point, separated by arbitrary whitespace with a multiple of
838 3 point count -->
839     <xs:simpleType name="ST PointsM3">
840         <xs:restriction base="xs:string">
841             <xs:whiteSpace value="collapse" />
842         <!--
843             <xs:pattern value="[rn][scs][rn]([rn][scs][rn]){2}((
844 [rn][scs][rn]){3})*"/>
845         -->
846         <xs:pattern value="((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-
847 9]+\(\.[\0-9]+\)?)( ?, ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ((\-
848 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-
849 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)){2}(( (\-|\+)?(([\0-9]+(\.[\0-
850 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( ?, ?)((\-|\+)?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-
851 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)){3})*" />
852         </xs:restriction>
853     </xs:simpleType>
854
855     <!-- EvenArray: List with even number of entries of non-negative numbers. -->
856     <xs:simpleType name="ST_EvenArrayPos">
857         <xs:restriction base="xs:string">
858             <xs:whiteSpace value="collapse" />
859     <!--
860         <xs:pattern value="[prn] [prn]([prn] [prn])*"/>
861     -->
862         <xs:pattern value="(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-
863 9]+\(\.[\0-9]+\)?)(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)( (\+?(([\0-9]+(\.[\0-
864 9]+\(\.[\0-9]+\)?)|(\.[\0-9]+))((e|E)(\-|\+)?[\0-9]+)?)(\+?(([\0-9]+(\.[\0-9]+)?)|(\.[\0-9]+))((e|E)(\
865 9]+\(\.[\0-9]+\)?)))*" />
866         </xs:restriction>
867     </xs:simpleType>
868
869     <!-- Array: List of numbers. -->

```

```

870     <xs:simpleType name="ST_Array">
871         <xs:restriction base="xs:string">
872             <xs:whiteSpace value="collapse" />
873     <!--
874         <xs:pattern value="([rn] ?)*"/>
875     -->
876         <xs:pattern value="(((\-|\+)?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-
877     |\+)?[0-9]+)? ?)*" />
878         </xs:restriction>
879     </xs:simpleType>
880
881     <!-- viewBox: 4 numbers, separated by , and arbitrary whitespace. Second number
882     pair must be non-negative -->
883     <xs:simpleType name="ST_ViewBox">
884         <xs:restriction base="xs:string">
885             <xs:whiteSpace value="collapse" />
886     <!--
887         <xs:pattern value="[rn][scs][rn][scs][prn][scs][prn]"/>
888     -->
889         <xs:pattern value="(((\-|\+)?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-
890     |\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?(
891     ?, ?)(\+?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)(\+?(([\-
892     9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />
893         </xs:restriction>
894     </xs:simpleType>
895
896     <!-- ContentBox: 4 non-negative numbers, separated by commas and arbitrary
897     whitespace -->
898     <xs:simpleType name="ST_ContentBox">
899         <xs:restriction base="xs:string">
900             <xs:whiteSpace value="collapse" />
901     <!--
902         <xs:pattern value="[prn][scs][prn][scs][prn][scs][prn]"/>
903     -->
904         <xs:pattern value="(\+?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-
905     9]+)?( ?, ?)(\+?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?,
906     ?)(\+?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)(\+?(([\-0-
907     9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />
908         </xs:restriction>
909     </xs:simpleType>
910
911     <!-- BleedBox: 4 numbers, separated by , and arbitrary whitespace. Second number
912     pair must be non-negative -->
913     <xs:simpleType name="ST_BleedBox">
914         <xs:restriction base="xs:string">
915             <xs:whiteSpace value="collapse" />
916     <!--
917         <xs:pattern value="[rn][scs][rn][scs][prn][scs][prn]"/>
918     -->
919         <xs:pattern value="(((\-|\+)?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-
920     |\+)?[0-9]+)?( ?, ?)((\-|\+)?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?(
921     ?, ?)(\+?(([\-0-9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)(\+?(([\-
922     9]+\.[\-0-9]+)?)|(\.[\-0-9]+))((e|E)(\-|\+)?[0-9]+)?)" />
923         </xs:restriction>
924     </xs:simpleType>

```





```

980         ((\([pint](:[pint])?\))?[uint])?\
981         (,[prn]?([rn]?([rn])?)?)?\
982     )" />
983 -->
984     <xs:pattern value="(((\((([1-9][0-9]*)(:([1-9][0-9]*)?)\))?)?([0-
985 9]+)?(,\( \+?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?(,( \-|\+)?((([0-
986 9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?(,( \-|\+)?((([0-9]+(\.[0-
987 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))?)?);(\((([1-9][0-9]*)(:([1-9][0-
988 9]*)?)\))?)?([0-9]+)?(,\( \+?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
989 9]+)?(,( \-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?(,( \-
990 |\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))?)?)" />
991     </xs:restriction>
992 </xs:simpleType>
993
994 <!-- UnicodeString grammar -->
995 <xs:simpleType name="ST_UnicodeString">
996     <xs:restriction base="xs:string">
997         <xs:pattern value="([\^{\}|\(\{\})|(\.|\[r\n])*)?[-.*]?"/>
998     </xs:restriction>
999 </xs:simpleType>
1000
1001 <!-- Abbreviated Geometry grammar for Path.Data , clip and Geometries -->
1002 <xs:simpleType name="ST_AbbrGeomF">
1003     <xs:restriction base="xs:string">
1004         <xs:whiteSpace value="collapse" />
1005 <!--
1006     <xs:pattern value="(F ?(0|1))?\
1007         ( ?(M|m)( ?[rn][scs][rn]))\
1008         (\
1009         ( ?(M|m)( ?[rn][scs][rn]))|\
1010         ( ?(L|l)( ?[rn][scs][rn])( [rn][scs][rn])*)|\
1011         ( ?(H|h|V|v)( ?[rn])( [rn])*)|\
1012         ( ?(Q|q|S|s)( ?[rn][scs][rn] [rn][scs][rn]))((
1013 [rn][scs][rn]){2})*|\
1014         ( ?(C|c)( ?[rn][scs][rn]( [rn][scs][rn]){2}))((
1015 [rn][scs][rn]){3})*|\
1016         ( ?(A|a)( ?[rn][scs][rn] [rn] [0-1] [0-1]
1017 [rn][scs][rn])\
1018         ( [rn][scs][rn] [rn] [0-1] [0-1]
1019 [rn][scs][rn])*)|\
1020         ( ?(Z|z))\
1021         )" />
1022 -->
1023     <xs:pattern value="(F ?(0|1))?( ?(M|m)( ?((\-|\+)?((([0-9]+(\.[0-
1024 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-
1025 9]+))((e|E)(\-|\+)?[0-9]+)?))(( ?(M|m)( ?((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-
1026 9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
1027 |\+)?[0-9]+)?))|( ?(L|l)( ?((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-
1028 9]+)?( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))(( \-
1029 |\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?((([0-
1030 9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))*)|( ?(H|h|V|v)( ?((\-|\+)?((([0-
1031 9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))(( \-|\+)?((([0-9]+(\.[0-
1032 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?))*)|( ?(Q|q|S|s)( ?((\-|\+)?((([0-9]+(\.[0-
1033 9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+)?( ?, ?)((\-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-
1034 9]+))((e|E)(\-|\+)?[0-9]+)?))(( \-|\+)?((([0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-

```

```

1035 9]+?) ( ?, ?) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ) ( ( ( \-
1036 | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-
1037 9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) { 2 } * ) | ( ? ( C | c ) ( ? ( \- | \+ ) ? ( ( [0-
1038 9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-
1039 9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1040 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \-
1041 | \+ ) ? [0-9]+ ) ? ) { 2 } ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) (
1042 ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) { 3 } * ) | ( ? ( A | a ) (
1043 ? ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-
1044 9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1045 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) [0-1] [0-1] ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1046 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \-
1047 | \+ ) ? [0-9]+ ) ? ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?,
1048 ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( \- | \+ ) ? ( ( [0-
1049 9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) [0-1] [0-1] ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-
1050 9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1051 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) * ) | ( ? ( Z | z ) ) * " / >
1052 < / x s : r e s t r i c t i o n >
1053 < / x s : s i m p l e T y p e >
1054
1055 < ! - - A b b r e v i a t e d G e o m e t r y g r a m m a r f o r P a t G e o m e t r y . F i g u r e s - - >
1056 < x s : s i m p l e T y p e n a m e = " S T _ A b b r G e o m " >
1057 < x s : r e s t r i c t i o n b a s e = " x s : s t r i n g " >
1058 < x s : w h i t e S p a c e v a l u e = " c o l l a p s e " / >
1059 < ! - -
1060 < x s : p a t t e r n v a l u e = " ( ? ( M | m ) ( ? [ r n ] [ s c s ] [ r n ] ) ) \
1061 ( \
1062 ( ? ( M | m ) ( ? [ r n ] [ s c s ] [ r n ] ) ) | \
1063 ( ? ( L | l ) ( ? [ r n ] [ s c s ] [ r n ] ) ( [ r n ] [ s c s ] [ r n ] ) * ) | \
1064 ( ? ( H | h | V | v ) ( ? [ r n ] ) ( [ r n ] ) * ) | \
1065 ( ? ( Q | q | S | s ) ( ? [ r n ] [ s c s ] [ r n ] [ r n ] [ s c s ] [ r n ] ) ( (
1066 [ r n ] [ s c s ] [ r n ] ) { 2 } * ) | \
1067 ( ? ( C | c ) ( ? [ r n ] [ s c s ] [ r n ] ( [ r n ] [ s c s ] [ r n ] ) { 2 } ) ( (
1068 [ r n ] [ s c s ] [ r n ] ) { 3 } * ) | \
1069 ( ? ( A | a ) ( ? [ r n ] [ s c s ] [ r n ] [ r n ] [ 0 - 1 ] [ 0 - 1 ]
1070 [ r n ] [ s c s ] [ r n ] ) \
1071 ( [ r n ] [ s c s ] [ r n ] [ r n ] [ 0 - 1 ] [ 0 - 1 ]
1072 [ r n ] [ s c s ] [ r n ] ) * ) | \
1073 ( ? ( Z | z ) ) \
1074 ) * " / >
1075 - - >
1076 < x s : p a t t e r n v a l u e = " ( ? ( M | m ) ( ? ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1077 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \-
1078 | \+ ) ? [0-9]+ ) ? ) ) ( ? ( M | m ) ( ? ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-
1079 9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ) | (
1080 ? ( L | l ) ( ? ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \-
1081 | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-
1082 9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1083 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) * ) | ( ? ( H | h | V | v ) ( ? ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-
1084 9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \-
1085 | \+ ) ? [0-9]+ ) ? ) * ) | ( ? ( Q | q | S | s ) ( ? ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \-
1086 | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? )
1087 ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-
1088 9]+ ( \. [0-9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ) ( ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-
1089 9]+ ) ? ) | ( \. [0-9]+ ) ) ( ( e | E ) ( \- | \+ ) ? [0-9]+ ) ? ) ( ?, ?) ( \- | \+ ) ? ( ( [0-9]+ ( \. [0-9]+ ) ? ) | ( \. [0-

```

```

1090 9+))((e|E)(\-|\+)?[0-9]+?)}){2})*|( ?(C|c)( ?((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-
1091 9+))((e|E)(\-|\+)?[0-9]+?) ( ?, ?)((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
1092 |\+)?[0-9]+?) ( (\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( ?,
1093 ?)((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?)}){2})*(( (\-
1094 |\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( ?, ?)((\-|\+)?((\.[0-
1095 9+)(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?)}){3})*|( ?(A|a)( ?((\-|\+)?((\.[0-
1096 9+)(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( ?, ?)((\-|\+)?((\.[0-9]+(\.[0-
1097 9+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( (\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-
1098 9+))((e|E)(\-|\+)?[0-9]+?) [0-1] [0-1] ((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-
1099 9+))((e|E)(\-|\+)?[0-9]+?) ( ?, ?)((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-
1100 |\+)?[0-9]+?) ( (\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( ?,
1101 ?)((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( (\-|\+)?((\.[0-
1102 9+)(\.[0-9]+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) [0-1] [0-1] ((\-|\+)?((\.[0-9]+(\.[0-
1103 9+)?)|(\.[0-9]+))((e|E)(\-|\+)?[0-9]+?) ( ?, ?)((\-|\+)?((\.[0-9]+(\.[0-9]+)?)|(\.[0-
1104 9+))((e|E)(\-|\+)?[0-9]+?)})*)|( ?(Z|z)))*" />
1105     </xs:restriction>
1106   </xs:simpleType>
1107
1108   <!-- Image reference via Uri -->
1109   <xs:simpleType name="ST_UriImage">
1110     <xs:restriction base="xs:anyURI">
1111       <xs:pattern value="([^\{\}.*]*)" />
1112     </xs:restriction>
1113   </xs:simpleType>
1114
1115   <!-- Image reference via ColorConvertedBitmap -->
1116   <xs:simpleType name="ST_CtxBmpImage">
1117     <xs:restriction base="xs:string">
1118       <xs:pattern value="\{ColorConvertedBitmap[\s]+[\S]+[\s]+[\S]+\}[\s]*" />
1119     </xs:restriction>
1120   </xs:simpleType>
1121
1122   <!-- Image reference via Uri or ColorConvertedBitmap -->
1123   <xs:simpleType name="ST_UriCtxBmp">
1124     <xs:union memberTypes="ST_UriImage ST_CtxBmpImage" />
1125   </xs:simpleType>
1126
1127   <!-- Resource reference -->
1128   <xs:simpleType name="ST_RscRef">
1129     <xs:restriction base="xs:string">
1130       <xs:pattern value="\{StaticResource[\s]+[\S]+\}[\s]*" />
1131     </xs:restriction>
1132   </xs:simpleType>
1133
1134   <!-- Resource reference OR Color -->
1135   <xs:simpleType name="ST_RscRefColor">
1136     <xs:union memberTypes="ST_Color ST_RscRef" />
1137   </xs:simpleType>
1138
1139   <!-- Resource reference OR Compact Matrix-->
1140   <xs:simpleType name="ST_RscRefMatrix">
1141     <xs:union memberTypes="ST_Matrix ST_RscRef" />
1142   </xs:simpleType>
1143
1144   <!-- Resource reference OR AbbrGeomF-->

```

```

1145 <xs:simpleType name="ST_RscRefAbbrGeomF">
1146   <xs:union memberTypes="ST_AbbrGeomF ST_RscRef" />
1147 </xs:simpleType>
1148
1149 <!-- Sweep Direction enumeration -->
1150 <xs:simpleType name="ST_SweepDirection">
1151   <xs:restriction base="xs:string">
1152     <xs:enumeration value="Clockwise" />
1153     <xs:enumeration value="Counterclockwise" />
1154   </xs:restriction>
1155 </xs:simpleType>
1156
1157 <!-- Dash Cap enumeration -->
1158 <xs:simpleType name="ST_DashCap">
1159   <xs:restriction base="xs:string">
1160     <xs:enumeration value="Flat" />
1161     <xs:enumeration value="Round" />
1162     <xs:enumeration value="Square" />
1163     <xs:enumeration value="Triangle" />
1164   </xs:restriction>
1165 </xs:simpleType>
1166
1167 <!-- Line Cap enumeration -->
1168 <xs:simpleType name="ST_LineCap">
1169   <xs:restriction base="xs:string">
1170     <xs:enumeration value="Flat" />
1171     <xs:enumeration value="Round" />
1172     <xs:enumeration value="Square" />
1173     <xs:enumeration value="Triangle" />
1174   </xs:restriction>
1175 </xs:simpleType>
1176
1177 <!-- Line Join enumeration -->
1178 <xs:simpleType name="ST_LineJoin">
1179   <xs:restriction base="xs:string">
1180     <xs:enumeration value="Miter" />
1181     <xs:enumeration value="Bevel" />
1182     <xs:enumeration value="Round" />
1183   </xs:restriction>
1184 </xs:simpleType>
1185
1186 <!-- Tile Mode enumeration -->
1187 <xs:simpleType name="ST_TileMode">
1188   <xs:restriction base="xs:string">
1189     <xs:enumeration value="None" />
1190     <xs:enumeration value="Tile" />
1191     <xs:enumeration value="FlipX" />
1192     <xs:enumeration value="FlipY" />
1193     <xs:enumeration value="FlipXY" />
1194   </xs:restriction>
1195 </xs:simpleType>
1196
1197 <!-- Color Interpolation Mode enumeration -->
1198 <xs:simpleType name="ST_ClrIntMode">
1199   <xs:restriction base="xs:string">

```

```
1200         <xs:enumeration value="ScRgbLinearInterpolation" />
1201         <xs:enumeration value="SRgbLinearInterpolation" />
1202     </xs:restriction>
1203 </xs:simpleType>
1204
1205 <!-- SpreadMethod Mode enumeration -->
1206 <xs:simpleType name="ST_SpreadMethod">
1207     <xs:restriction base="xs:string">
1208         <xs:enumeration value="Pad" />
1209         <xs:enumeration value="Reflect" />
1210         <xs:enumeration value="Repeat" />
1211     </xs:restriction>
1212 </xs:simpleType>
1213
1214 <!-- FillRule Mode enumeration -->
1215 <xs:simpleType name="ST_FillRule">
1216     <xs:restriction base="xs:string">
1217         <xs:enumeration value="EvenOdd" />
1218         <xs:enumeration value="NonZero" />
1219     </xs:restriction>
1220 </xs:simpleType>
1221
1222 <!-- Edge Mode enumeration -->
1223 <xs:simpleType name="ST_EdgeMode">
1224     <xs:restriction base="xs:string">
1225         <xs:enumeration value="Aliased" />
1226     </xs:restriction>
1227 </xs:simpleType>
1228
1229 <!-- Style Simulation Enumeration -->
1230 <xs:simpleType name="ST_StyleSimulations">
1231     <xs:restriction base="xs:string">
1232         <xs:enumeration value="None" />
1233         <xs:enumeration value="ItalicSimulation" />
1234         <xs:enumeration value="BoldSimulation" />
1235         <xs:enumeration value="BoldItalicSimulation" />
1236     </xs:restriction>
1237 </xs:simpleType>
1238
1239 <!-- ViewUnits Enumeration -->
1240 <xs:simpleType name="ST_ViewUnits">
1241     <xs:restriction base="xs:string">
1242         <xs:enumeration value="Absolute" />
1243     </xs:restriction>
1244 </xs:simpleType>
1245
1246 <!-- MappingMode Enumeration -->
1247 <xs:simpleType name="ST_MappingMode">
1248     <xs:restriction base="xs:string">
1249         <xs:enumeration value="Absolute" />
1250     </xs:restriction>
1251 </xs:simpleType>
1252 </xs:schema>
```









## 1 A.4 Document Structure

2 The schema shown below is also provided in electronic form as a file named  
 3 [OpenXPSDocumentStructure.xsd](#), which is contained in an accompanying zip archive named  
 4 "[OpenXPS WC3 Schemas.zip](#)". If discrepancies exist between the representation as published  
 5 below and the corresponding electronic version, the published version below is the definitive  
 6 version.

```

7
8 <?xml version="1.0" encoding="UTF-8"?><xs:schema
9 targetNamespace="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/
10 documentstructure"
11 xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentst
12 ructure" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
13 blockDefault="#all">
14
15     <xs:import namespace="http://www.w3.org/XML/1998/namespace" />
16
17     <!-- =====DocumentStructure Part===== -->
18     <!-- Complex Types -->
19     <xs:complexType name="CT_DocumentStructure">
20         <xs:sequence>
21             <xs:element ref="DocumentStructure.Outline" minOccurs="0" />
22             <xs:element ref="Story" minOccurs="0" maxOccurs="unbounded" />
23         </xs:sequence>
24     </xs:complexType>
25     <xs:complexType name="CT_CP_Outline">
26         <xs:sequence>
27             <xs:element ref="DocumentOutline" />
28         </xs:sequence>
29     </xs:complexType>
30     <xs:complexType name="CT_DocumentOutline">
31         <xs:sequence>
32             <xs:element ref="OutlineEntry" maxOccurs="unbounded" />
33         </xs:sequence>
34         <xs:attributeGroup ref="AG_DocumentOutline" />
35     </xs:complexType>
36     <xs:complexType name="CT_OutlineEntry">
37         <xs:attributeGroup ref="AG_OutlineEntry" />
38     </xs:complexType>
39     <xs:complexType name="CT_Story">
40         <xs:sequence>
41             <xs:element ref="StoryFragmentReference" maxOccurs="unbounded" />
42         </xs:sequence>
43         <xs:attributeGroup ref="AG_Story" />
44     </xs:complexType>
45     <xs:complexType name="CT_StoryFragmentReference">
46         <xs:attributeGroup ref="AG_StoryFragmentReference" />
47     </xs:complexType>
48     <!-- Simple Types -->
49     <!-- A Name (ID with pattern restriction according to OpenXPS spec) -->
50     <xs:simpleType name="ST_Name">

```

```

44     <xs:restriction base="xs:string">
45         <xs:pattern
46 value="(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl})(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl}|\p{Mn}|\
47 p{Mc}|\p{Nd}|\p{Lm}|_)*" />
48     </xs:restriction>
49 </xs:simpleType>
50 <!-- A Unique Name (ID with pattern restriction according to OpenXPS spec) -->
51 <xs:simpleType name="ST_NameUnique">
52     <xs:restriction base="xs:ID">
53         <xs:pattern
54 value="(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl})(\p{Lu}|\p{Ll}|\p{Lo}|\p{Lt}|\p{Nl}|\p{Mn}|\
55 p{Mc}|\p{Nd}|\p{Lm}|_)*" />
56     </xs:restriction>
57 </xs:simpleType>
58 <!-- integer greater than or equal to 1 inclusive -->
59 <xs:simpleType name="ST_IntGEOne">
60     <xs:restriction base="xs:int">
61         <xs:minInclusive value="1" />
62     </xs:restriction>
63 </xs:simpleType>
64 <!-- Elements -->
65 <xs:element name="DocumentStructure" type="CT_DocumentStructure">
66     </xs:element>
67 <xs:element name="DocumentStructure.Outline" type="CT_CP_Outline">
68     </xs:element>
69 <xs:element name="DocumentOutline" type="CT_DocumentOutline">
70     </xs:element>
71 <xs:element name="OutlineEntry" type="CT_OutlineEntry">
72     </xs:element>
73 <xs:element name="Story" type="CT_Story">
74     </xs:element>
75 <xs:element name="StoryFragmentReference" type="CT_StoryFragmentReference">
76     </xs:element>
77 <!-- Attribute Groups -->
78 <xs:attributeGroup name="AG_DocumentOutline">
79     <xs:attribute ref="xml:lang" use="required">
80         </xs:attribute>
81 </xs:attributeGroup>
82 <xs:attributeGroup name="AG_OutlineEntry">
83     <xs:attribute name="OutlineLevel" type="ST_IntGEOne" use="optional"
84 default="1">
85         </xs:attribute>
86     <xs:attribute name="OutlineTarget" type="xs:anyURI" use="required">
87         </xs:attribute>
88     <xs:attribute name="Description" type="xs:string" use="required">
89         </xs:attribute>
90     <xs:attribute ref="xml:lang" use="optional">
91         </xs:attribute>
92 </xs:attributeGroup>
93 <xs:attributeGroup name="AG_Story">
94     <xs:attribute name="StoryName" type="xs:string" use="required">
95         </xs:attribute>
96 </xs:attributeGroup>
97 <xs:attributeGroup name="AG_StoryFragmentReference">
98     <xs:attribute name="FragmentName" type="xs:string" use="optional">

```

```

99         </xs:attribute>
100         <xs:attribute name="Page" type="ST_IntGEOne" use="required">
101             </xs:attribute>
102     </xs:attributeGroup>
103     <!-- =====StoryFragments Part===== -->
104     <!-- Complex Types -->
105     <xs:complexType name="CT_StoryFragments">
106         <xs:sequence>
107             <xs:element ref="StoryFragment" maxOccurs="unbounded" />
108         </xs:sequence>
109     </xs:complexType>
110     <xs:complexType name="CT_StoryFragment">
111         <xs:sequence>
112             <xs:element ref="StoryBreak" minOccurs="0" />
113             <xs:choice maxOccurs="unbounded">
114                 <xs:element ref="SectionStructure" />
115                 <xs:element ref="ParagraphStructure" />
116                 <xs:element ref="ListStructure" />
117                 <xs:element ref="TableStructure" />
118                 <xs:element ref="FigureStructure" />
119             </xs:choice>
120             <xs:element ref="StoryBreak" minOccurs="0" />
121         </xs:sequence>
122         <xs:attributeGroup ref="AG_StoryFragment" />
123     </xs:complexType>
124     <xs:complexType name="CT_Break">
125         </xs:complexType>
126     <xs:complexType name="CT_Section">
127         <xs:choice maxOccurs="unbounded">
128             <xs:element ref="ParagraphStructure" />
129             <xs:element ref="ListStructure" />
130             <xs:element ref="TableStructure" />
131             <xs:element ref="FigureStructure" />
132         </xs:choice>
133     </xs:complexType>
134     <xs:complexType name="CT_Paragraph">
135         <xs:choice minOccurs="0" maxOccurs="unbounded">
136             <xs:element ref="NamedElement" />
137         </xs:choice>
138     </xs:complexType>
139     <xs:complexType name="CT_Table">
140         <xs:choice maxOccurs="unbounded">
141             <xs:element ref="TableRowGroupStructure" />
142         </xs:choice>
143     </xs:complexType>
144     <xs:complexType name="CT_TableRowGroup">
145         <xs:choice maxOccurs="unbounded">
146             <xs:element ref="TableRowStructure" />
147         </xs:choice>
148     </xs:complexType>
149     <xs:complexType name="CT_TableRow">
150         <xs:choice maxOccurs="unbounded">
151             <xs:element ref="TableCellStructure" />
152         </xs:choice>
153     </xs:complexType>

```

```

154 <xs:complexType name="CT_TableCell">
155   <xs:choice minOccurs="0" maxOccurs="unbounded">
156     <xs:element ref="ParagraphStructure" />
157     <xs:element ref="ListStructure" />
158     <xs:element ref="TableStructure" />
159     <xs:element ref="FigureStructure" />
160   </xs:choice>
161   <xs:attributeGroup ref="AG_TableCell" />
162 </xs:complexType>
163 <xs:complexType name="CT_List">
164   <xs:choice maxOccurs="unbounded">
165     <xs:element ref="ListItemStructure" />
166   </xs:choice>
167 </xs:complexType>
168 <xs:complexType name="CT_ListItem">
169   <xs:choice minOccurs="0" maxOccurs="unbounded">
170     <xs:element ref="ParagraphStructure" />
171     <xs:element ref="ListStructure" />
172     <xs:element ref="TableStructure" />
173     <xs:element ref="FigureStructure" />
174   </xs:choice>
175   <xs:attributeGroup ref="AG_ListItem" />
176 </xs:complexType>
177 <xs:complexType name="CT_Figure">
178   <xs:choice minOccurs="0" maxOccurs="unbounded">
179     <xs:element ref="NamedElement" />
180   </xs:choice>
181 </xs:complexType>
182 <xs:complexType name="CT_NamedElement">
183   <xs:attributeGroup ref="AG_NamedElement" />
184 </xs:complexType>
185 <!-- Simple Types -->
186 <!-- FragmentType enumeration -->
187 <xs:simpleType name="ST_FragmentType">
188   <xs:restriction base="xs:string">
189     <xs:enumeration value="Content" />
190     <xs:enumeration value="Header" />
191     <xs:enumeration value="Footer" />
192   </xs:restriction>
193 </xs:simpleType>
194 <xs:simpleType name="ST_Location">
195 <xs:restriction base="xs:string">
196 <xs:pattern value="([0-9][0-9]*)(\,[0-9][0-9]*)*" />
197 </xs:restriction>
198 </xs:simpleType>
199 <xs:simpleType name="ST_TableSpan">
200   <xs:restriction base="xs:int">
201     <xs:minInclusive value="1" />
202   </xs:restriction>
203 </xs:simpleType>
204 <xs:simpleType name="ST_ElementIndex">
205 <xs:restriction base="xs:int">
206 <xs:minInclusive value="0" />
207 </xs:restriction>
208 </xs:simpleType>

```

```

209 <!-- Elements -->
210 <xs:element name="StoryFragments" type="CT_StoryFragments">
211     </xs:element>
212 <xs:element name="StoryFragment" type="CT_StoryFragment">
213     </xs:element>
214 <xs:element name="StoryBreak" type="CT_Break">
215     </xs:element>
216 <xs:element name="SectionStructure" type="CT_Section">
217     </xs:element>
218 <xs:element name="ParagraphStructure" type="CT_Paragraph">
219     </xs:element>
220 <xs:element name="TableStructure" type="CT_Table">
221     </xs:element>
222 <xs:element name="TableRowGroupStructure" type="CT_TableRowGroup">
223     </xs:element>
224 <xs:element name="TableRowStructure" type="CT_TableRow">
225     </xs:element>
226 <xs:element name="TableCellStructure" type="CT_TableCell">
227     </xs:element>
228 <xs:element name="ListStructure" type="CT_List">
229     </xs:element>
230 <xs:element name="ListItemStructure" type="CT_ListItem">
231     </xs:element>
232 <xs:element name="FigureStructure" type="CT_Figure">
233     </xs:element>
234 <xs:element name="NamedElement" type="CT_NamedElement">
235     </xs:element>
236 <!-- Attribute Groups -->
237 <xs:attributeGroup name="AG_StoryFragment">
238     <xs:attribute name="StoryName" type="xs:string" use="optional">
239         </xs:attribute>
240     <xs:attribute name="FragmentName" type="xs:string" use="optional">
241         </xs:attribute>
242     <xs:attribute name="FragmentType" type="ST_FragmentType" use="required">
243         </xs:attribute>
244 </xs:attributeGroup>
245 <xs:attributeGroup name="AG_TableCell">
246     <xs:attribute name="RowSpan" type="ST_TableSpan" use="optional"
247 default="1">
248         </xs:attribute>
249     <xs:attribute name="ColumnSpan" type="ST_TableSpan" use="optional"
250 default="1">
251         </xs:attribute>
252 </xs:attributeGroup>
253 <xs:attributeGroup name="AG_ListItem">
254     <xs:attribute name="Marker" type="ST_NameUnique" use="optional">
255         </xs:attribute>
256 </xs:attributeGroup>
257 <xs:attributeGroup name="AG_NamedElement">
258     <xs:attribute name="NameReference" type="ST_Name" use="required">
259         </xs:attribute>
260 </xs:attributeGroup>
261 </xs:schema>

```



## 1 **A.5 Discard Control**

2 The schema shown below is also provided in electronic form as a file named  
3 [OpenXPSDiscardControl.xsd](#), which is contained in an accompanying zip archive named  
4 "[OpenXPS WC3 Schemas.zip](#)". If discrepancies exist between the representation as published  
5 below and the corresponding electronic version, the published version below is the definitive  
6 version.

```
7  
1 <?xml version="1.0" encoding="UTF-8"?>  
2 <xs:schema  
3 targetNamespace="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/  
4 discard-control"  
5 xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/discard-  
6 control" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"  
7 blockDefault="#all">  
8  
9     <xs:complexType name="CT_DiscardControl">  
10         <xs:sequence>  
11             <xs:element ref="Discard" minOccurs="0" maxOccurs="unbounded" />  
12         </xs:sequence>  
13     </xs:complexType>  
14  
15     <xs:complexType name="CT_Discard">  
16         <xs:attribute name="SentinelPage" type="xs:anyURI" use="required">  
17         </xs:attribute>  
18         <xs:attribute name="Target" type="xs:anyURI" use="required">  
19         </xs:attribute>  
20     </xs:complexType>  
21  
22     <xs:element name="DiscardControl" type="CT_DiscardControl">  
23     </xs:element>  
24  
25     <xs:element name="Discard" type="CT_Discard">  
26     </xs:element>  
27  
28 </xs:schema>
```

## 1 A.6 3D-Graphic Content

2 The schema shown below is also provided in electronic form as a file named OpenXPS3D.xsd,  
 3 which is contained in an accompanying zip archive named "OpenXPS 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
1 <?xml version="1.0" encoding="utf-8"?>
2 <!-- OpenXPS 3D Graphics content Schema prepared by Nathan Crews - Autodesk -->
3 <xs:schema xmlns="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0"
4
5 xmlns:oxps1="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
6 xmlns:mstns="http://tempuri.org/XMLSchema.xsd"
7 xmlns:xs="http://www.w3.org/2001/XMLSchema"
8
9 xmlns:x="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resou
10 rcedictionary-key"
11 targetNamespace="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0"
12 elementFormDefault="qualified" blockDefault="#all">
13 <!-- Import OpenXPS and related XML Schemas -->
14 <xs:import
15 namespace="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"/>
16 <xs:import namespace=
17
18 "http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedicti
19 onary-key"/>
20 <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
21 <!-- Names used for types and groups follows XPS 1.0 pattern:
22 ST_* simpleType
23 CT_* complexType
24 G_* group
25 AG_* attributeGroup -->
26 <!-- Complex Types -->
27 <xs:complexType name="CT_Brush3D">
28 <xs:sequence>
29 <xs:element ref="oxps1:ImageBrush.Transform" minOccurs="0"/>
30 </xs:sequence>
31 <xs:attributeGroup ref="AG_Brush3D"/>
32 </xs:complexType>
33 <!-- Root elements -->
34 <xs:element name="Brush3D">
35 <xs:complexType>
36 <xs:complexContent>
37 <xs:extension base="CT_Brush3D"/>
38 </xs:complexContent>
39 </xs:complexType>
40 </xs:element>
41 <!-- Attribute Groups -->
42 <xs:attributeGroup name="AG_Brush3D">
43 <xs:attribute name="Source3D" type="ST_UriImage3D" use="required"/>
44 <xs:attribute ref="x:Key"/>
45 <xs:attribute name="Transform" type="oxps1:ST_RscRefMatrix"/>

```



```
46 <xs:attribute name="Viewbox" type="oxps1:ST_ViewBox" use="required"/>
47 <xs:attribute name="Viewport" type="oxps1:ST_ViewBox" use="required"/>
48 <xs:attribute name="ViewboxUnits" type="oxps1:ST_ViewUnits" use="required"
49     fixed="Absolute"/>
50 <xs:attribute name="ViewportUnits" type="oxps1:ST_ViewUnits" use="required"
51     fixed="Absolute"/>
52 </xs:attributeGroup>
53 <!-- Simple Types -->
54 <xs:simpleType name="ST_UriImage3D">
55     <xs:restriction base="xs:anyURI">
56         <xs:pattern value="([^\{].*)?"/>
57     </xs:restriction>
58 </xs:simpleType>
59 </xs:schema>
```



## 1 **B. Schemas – RELAX NG**

2 **This annex is informative**

---

### 3 **B.1 Signature Definitions**

4 The schema shown below is also provided in electronic form as a file named  
5 [OpenXPS](#)SignatureDefinitions.rnc, which is contained in an accompanying zip archive named  
6 "[OpenXPS](#) RELAX NG Schemas.zip". If discrepancies exist between the representation as  
7 published below and the corresponding electronic version, the published version below is the  
8 definitive version.

9 <<Schema goes here>>



---

## 1 **B.2 OpenXPS Document**

2 The schema shown below is also provided in electronic form as a file named  
3 [OpenXPSDocument.rnc](#), which is contained in an accompanying zip archive named "[OpenXPS](#)  
4 [RELAX NG Schemas.zip](#)". If discrepancies exist between the representation as published below  
5 and the corresponding electronic version, the published version below is the definitive version.

6 <<Schema goes here>>



---

### 1 **B.3 Resource Dictionary Key**

2 The schema shown below is also provided in electronic form as a file named  
3 [OpenXPSResourceDictionaryKey.rnc](#), which is contained in an accompanying zip archive named  
4 "[OpenXPS RELAX NG Schemas.zip](#)". If discrepancies exist between the representation as  
5 published below and the corresponding electronic version, the published version below is the  
6 definitive version.

7 <<Schema goes here>>





---

## 1 **B.4 Document Structure**

2 The schema shown below is also provided in electronic form as a file named  
3 [OpenXPS](#)DocumentStructure.rnc, which is contained in an accompanying zip archive named  
4 "[OpenXPS RELAX NG Schemas.zip](#)". If discrepancies exist between the representation as  
5 published below and the corresponding electronic version, the published version below is the  
6 definitive version.

7 <<Schema goes here>>



---

## 1 **B.5 Discard Control**

2 The schema shown below is also provided in electronic form as a file named  
3 [OpenXPSDiscardControl.rnc](#)~~xsd~~, which is contained in an accompanying zip archive named  
4 "[OpenXPS RELAX NG Schemas.zip](#)". If discrepancies exist between the representation as  
5 published below and the corresponding electronic version, the published version below is the  
6 definitive version.

7 <<Schema goes here>>

---

## 1 **B.6 3D-Graphic Content**

2 The schema shown below is also provided in electronic form as a file named OpenXPS3D.xsd,  
3 | which is contained in an accompanying zip archive named "[OpenXPS RELAX NG Schemas.zip](#)".  
4 If discrepancies exist between the representation as published below and the corresponding  
5 electronic version, the published version below is the definitive version.

1 <<Schema goes here>>

2 **End of informative text.**

3

## 1 C. 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:
59      Read relative coordinate pair DX,DY
60      Let CURRENTPOINT.X = CURRENTPOINT.X + DX
  
```

```

1         Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
2         Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
3         if ( next character is not a letter )
4         {
5             GOTO label_1
6         }
7     }
8     else if ( CH == 'L' )
9     {
10        Create new PolyLineSegment S
11        Add S to CURRENTPATHFIGURE
12    label_2:
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_2
20        }
21    }
22    else if ( CH == 'h' )
23    {
24        Create new PolyLineSegment S
25        Add S to CURRENTPATHFIGURE
26    label_3:
27        Read relative coordinate value DX
28        Let CURRENTPOINT.X = CURRENTPOINT.X + DX
29        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
30        if ( next character is not a letter )
31        {
32            GOTO label_3
33        }
34    }
35    else if ( CH == 'H' )
36    {
37        Create new PolyLineSegment S
38        Add S to CURRENTPATHFIGURE
39    label_4:
40        Read coordinate value X
41        Let CURRENTPOINT.X = X
42        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
43        if ( next character is not a letter )
44        {
45            GOTO label_4
46        }
47    }
48    else if ( CH == 'v' )
49    {
50        Create new PolyLineSegment S
51        Add S to CURRENTPATHFIGURE
52    label_5:
53        Read relative coordinate value DY
54        Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
55        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
56        if ( next character is not a letter )
57        {
58            GOTO label_5
59        }
60    }
61    else if ( CH == 'V' )
62    {
63        Create new PolyLineSegment S
64        Add S to CURRENTPATHFIGURE
65    label_6:
66        Read coordinate value Y
67        Let CURRENTPOINT.Y = Y
68        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
69        if ( next character is not a letter )

```

```

1      {
2          GOTO label_6
3      }
4  }
5  else if ( CH == 'c' )
6  {
7      Create new PolyBezierSegment S
8      Add S to CURRENTPATHFIGURE
9  label_7:
10     Read relative coordinate pair DX,DY
11     Let POINT.X = CURRENTPOINT.X + DX
12     Let POINT.Y = CURRENTPOINT.Y + DY
13     Add POINT.X, POINT.Y to end of S.Points attribute list
14     Read coordinate pair DX,DY
15     Let POINT.X = CURRENTPOINT.X + DX
16     Let POINT.Y = CURRENTPOINT.Y + DY
17     Add POINT.X, POINT.Y to end of S.Points attribute list
18     Read coordinate pair DX,DY
19     Let CURRENTPOINT.X = CURRENTPOINT.X + DX
20     Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
21     Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
22     if ( next character is not a letter )
23     {
24         GOTO label_7
25     }
26 }
27 else if ( CH == 'C' )
28 {
29     Create new PolyBezierSegment S
30     Add S to CURRENTPATHFIGURE
31 label_8:
32     Read coordinate pair X,Y
33     Let POINT.X = X
34     Let POINT.Y = Y
35     Add POINT.X, POINT.Y to end of S.Points attribute list
36     Read coordinate pair X,Y
37     Let POINT.X = X
38     Let POINT.Y = Y
39     Add POINT.X, POINT.Y to end of S.Points attribute list
40     Read coordinate pair X,Y
41     Let CURRENTPOINT.X = X
42     Let CURRENTPOINT.Y = Y
43     Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
44     if ( next character is not a letter )
45     {
46         GOTO label_8
47     }
48 }
49 else if ( CH == 'q' )
50 {
51     Create new PolyQuadraticBezierSegment S
52     Add S to CURRENTPATHFIGURE
53 label_9:
54     Read relative coordinate pair DX,DY
55     Let POINT.X = CURRENTPOINT.X + DX
56     Let POINT.Y = CURRENTPOINT.Y + DY
57     Add POINT.X, POINT.Y to end of S.Points attribute list
58     Read relative coordinate pair DX,DY
59     Let CURRENTPOINT.X = CURRENTPOINT.X + DX
60     Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
61     Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
62     if ( next character is not a letter )
63     {
64         GOTO label_9
65     }
66 }
67 else ( if CH == 'Q' )
68 {

```

```

1         Create new PolyQuadraticBezierSegment S
2         Add S to CURRENTPATHFIGURE
3     label_10:
4         Read coordinate pair X,Y
5         Let POINT.X = X
6         Let POINT.Y = Y
7         Add POINT.X, POINT.Y to end of S.Points attribute list
8         Read coordinate pair X,Y
9         Let CURRENTPOINT.X = X
10        Let CURRENTPOINT.Y = Y
11        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
12        if ( next character is not a letter )
13            {
14                GOTO label_10
15            }
16        }
17        else if ( CH == 's' )
18            {
19                Create new PolyBezierSegment S
20                Add S to CURRENTPATHFIGURE
21        label_11:
22            if ( S.Points is non-empty )
23                {
24                    Let LASTCTRLPOINT = Point before last point in S.Points
25                    Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
26                    Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
27                }
28            else if ( segment before CURRENTPATHSEGMENT is a PolyBezierSegment )
29                {
30                    Let LASTCTRLPOINT = Point before last point in previous PolyBezierSegment
31                    Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
32                    Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
33                }
34            else
35                {
36                Let POINT = CURRENTPOINT
37                }
38
39            Add POINT.X, POINT.Y to end of S.Points attribute list
40            Read relative coordinate pair DX,DY
41            Let POINT.X = CURRENTPOINT.X + DX
42            Let POINT.Y = CURRENTPOINT.Y + DY
43            Add POINT.X, POINT.Y to end of S.Points attribute list
44            Read relative coordinate pair DX,DY
45            Let CURRENTPOINT.X = CURRENTPOINT.X + DX
46            Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
47            Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
48            if ( next character is not a letter )
49                {
50                GOTO label_11
51                }
52            }
53        else if ( CH == 'S' )
54            {
55                Create new PolyBezierSegment S
56                Add S to CURRENTPATHFIGURE
57        label_12:
58            if ( S.Points is non-empty )
59                {
60                    Let LASTCTRLPOINT = Point before last point in S.Points
61                    Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
62                    Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
63                }
64            else if ( segment before CURRENTPATHSEGMENT is a PolyBezierSegment )
65                {
66                    Let LASTCTRLPOINT = S.Point before last point in previous PolyBezierSegment
67                    Let POINT.X = 2 * CURRENTPOINT.X - LASTCTRLPOINT.X
68                    Let POINT.Y = 2 * CURRENTPOINT.Y - LASTCTRLPOINT.Y
69                }

```



```

1         else
2         {
3             Let POINT = CURRENTPOINT
4         }
5
6         Add POINT.X, POINT.Y to end of S.Points attribute list
7         Read coordinate pair X,Y
8         Let POINT.X = X
9         Let POINT.Y = Y
10        Add POINT.X, POINT.Y to end of S.Points attribute list
11        Read coordinate pair X,Y
12        Let CURRENTPOINT.X = X
13        Let CURRENTPOINT.Y = Y
14        Add CURRENTPOINT.X,CURRENTPOINT.Y to end of S.Points attribute list
15        if ( next character is not a letter )
16        {
17            GOTO label_12
18        }
19    }
20    else if ( CH == 'a' )
21    {
22        label_13:
23            Create new ArcSegment S
24            Add S to CURRENTPATHFIGURE
25            Read Radius Pair RX,RY
26            Read Rotation ROT
27            Read integer FLAG1
28            Read integer FLAG2
29            Read relative coordinate pair DX,DY
30            Let CURRENTPOINT.X = CURRENTPOINT.X + DX
31            Let CURRENTPOINT.Y = CURRENTPOINT.Y + DY
32            Let S.Point = CURRENTPOINT.X,CURRENTPOINT.Y
33            Let S.IsLargeArc = (FLAG1 == 1 ? true : false)
34            Let S.SweepDirection = (FLAG2 == 1 ? Clockwise : Counterclockwise)
35            Let S.RotationAngle = ROT
36            Let S.Size = RX, RY
37            if ( next character is not a letter )
38            {
39                GOTO label_13
40            }
41        }
42    else if ( CH == 'A' )
43    {
44        label_14:
45            Create new ArcSegment S
46            Add S to CURRENTPATHFIGURE
47            Read Radius Pair RX,RY
48            Read Rotation ROT
49            Read integer FLAG1
50            Read integer FLAG2
51            Read coordinate pair X,Y
52            Let CURRENTPOINT.X = X
53            Let CURRENTPOINT.Y = Y
54            Let S.Point = CURRENTPOINT.X,CURRENTPOINT.Y
55            Let S.IsLargeArc = (FLAG1 == 1 ? true : false)
56            Let S.SweepDirection = (FLAG2 == 1 ? Clockwise : Counterclockwise)
57            Let S.RotationAngle = ROT
58            Let S.Size = RX, RY
59            if ( next character is not a letter )
60            {
61                GOTO label_14
62            }
63        }
64    else if ( CH == 'z' or CH == 'Z' )
65    {
66        Let attribute CURRENTPATHFIGURE.IsClosed = true
67        Let CURRENTPOINT = First point of first segment of CURRENTPATHFIGURE
68        Let CURRENTPATHFIGURE = undefined
69    }

```

```
1      /* This case can not occur, because the input is assumed to be well-formed according to the markup
2      schema
3      else
4      {
5          ERROR: Invalid input character
6      }
7      */
8
9      if ( End of input reached )
10     {
11         Terminate algorithm, return PG
12     }
13     else
14     {
15         GOTO label_repeat
16     }
```

## 1 D. Standard Namespaces and Content Types

2 The following tables list the namespaces and content types used in OpenXPS packages and  
3 OpenXPS Documents.

### 4 D.1 XML Namespace URIs

5 *Table D-1. Package-wide namespaces*

Description	Namespace URI
Content Types	<a href="http://schemas.openxmlformats.org/package/2006/content-types">http://schemas.openxmlformats.org/package/2006/content-types</a>
Core Properties	<a href="http://schemas.openxmlformats.org/package/2006/metadata/core-properties">http://schemas.openxmlformats.org/package/2006/metadata/core-properties</a>
Digital Signatures	<a href="http://schemas.openxmlformats.org/package/2006/digital-signature">http://schemas.openxmlformats.org/package/2006/digital-signature</a>
Relationships	<a href="http://schemas.openxmlformats.org/package/2006/relationships">http://schemas.openxmlformats.org/package/2006/relationships</a>
Markup Compatibility	<a href="http://schemas.openxmlformats.org/markup-compatibility/2006">http://schemas.openxmlformats.org/markup-compatibility/2006</a>

6 *Table D-2. OpenXPS Document namespaces*

Description	Namespace URI
DiscardControl	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/discard-control">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/discard-control</a>
Document Structure	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure</a>
FixedDocument	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0</a>
FixedDocumentSequence	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0</a>
FixedPage	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0</a>
Resource Dictionary (Key attribute)	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedictionary-key">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/resourcedictionary-key</a>
Signature Definitions	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/signature-definitions">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/signature-definitions</a>
Story Fragments	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure</a>
targetNamespace	<a href="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0">http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0</a>
3D Graphics Content	<a href="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0">http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0</a>

## D.2 Content Types

The content types in the tables below MUST [be used by producers without parameters](#) ~~NOT include parameters~~ [M12.8]. If a consumer encounters the presence of parameters on these content types when the affected part is accessed it MUST instantiate an error condition [M12.7].

Table D-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
<a href="#">SignatureDefinitions</a>	<a href="#">application/vnd.ms-package.xps-signaturedefinitions+xml</a>

Table D-4. OpenXPS 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
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
<del>Windows Media Photo</del> <a href="#">JPEG XR</a> image	image/vnd.ms-photo
X3D image unicode	model/x3d+xml
X3D image binary encoding	model/x3d+binary

### 1 D.3 Relationship Types

2 *Table D-5. Package-wide relationship types*

Description	Relationship type
Core Properties	<a href="http://schemas.openxmlformats.org/package/2006/relationships/metadata/core-properties">http://schemas.openxmlformats.org/package/2006/relationships/metadata/core-properties</a>
Digital Signature	<a href="http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/signature">http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/signature</a>
Digital Signature Certificate	<a href="http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/certificate">http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/certificate</a>
Digital Signature Origin	<a href="http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/origin">http://schemas.openxmlformats.org/package/2006/relationships/digital-signature/origin</a>
Thumbnail	<a href="http://schemas.openxmlformats.org/package/2006/relationships/metadata/thumbnail">http://schemas.openxmlformats.org/package/2006/relationships/metadata/thumbnail</a>

3 *Table D-6. OpenXPS Document relationship types*

Description	Relationship type
Digital Signature Definitions	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/signature-definitions">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/signature-definitions</a>
DiscardControl	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/discard-control">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/discard-control</a>
DocumentStructure	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/documentstructure</a>
PrintTicket	<a href="http://schemas.openxps.org/oxps/v1.0/printticket">http://schemas.openxps.org/oxps/v1.0/printticket</a>
Required Resource	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/required-resource">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/required-resource</a>
Restricted Font	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/restricted-font">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/restricted-font</a>
StartPart	<a href="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/startpart">http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0/startpart</a>

---

<b>Description</b>	<b>Relationship type</b>
StoryFragments	xedrepresentation <a href="http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0/storyfragments">http://schemas.microsoft.com/xps/2005/06/schemas.openxps.org/oxps/v1.0/storyfragments</a>

---

## **E. Recommended File Name Extension and Content Types**

This annex provides details for implementations and external systems that need to identify OpenXPS Documents.

---

### **E.1 Identification of OpenXPS Documents**

Implementations are anticipated for multiple operating systems, including operating systems that use the concept of filename extension and/or content type to identify the format of files for processing. When required by such systems, and to enable interoperability with such systems, implementations SHOULD use a filename extension or termination sequence of .oxps and a content type of `application/oxps` [S14.1].

To avoid conflicts between OpenXPS Documents defined in this Standard and legacy formats, producers MUST NOT create OpenXPS Documents with filenames that end in the uppercase, lowercase, or mixed-case sequence .xps [M14.1]. Implementations are encouraged not to use `application/vnd.ms-xpsdocument` as a content type.

---

### **E.2 Embedding Producer Identification**

Producers SHOULD include an XML comment immediately following the start-tag of the FixedPage element. This comment SHOULD include details of the organization, product, and version that created the content with the intention that this could be used as an aid in the diagnosis of problem content [S14.2].

[Example:

```
<FixedPage Width="816" Height="1056"
  xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
  xml:lang="en-US">
  <!--Generated by: Henri Fazy A.G, Papon Café, Version: 3.14159265 -->
  <Glyphs Fill="#ff000000"
    FontUri="/Documents/1/Resources/Fonts/times.ttf" FontRenderingEmSize="12"
    OriginX="348" OriginY="106.4" UnicodeString="Good food indeed." />
</FixedPage>
```

end example]

**This subclause is informative**

---

### **E.3 Determination of OPC payload**

OpenXPS Documents follow requirements set out in this Standard, as well as other normative references including the Open Packaging Conventions. Implementations may use the following steps to identify an unknown file or stream as an OpenXPS Document within an Open Packaging Conventions payload.

1. Test that the file or stream is a ZIP file
  - a. As required by §9.2 of the Open Packaging Conventions

1 [b. First four bytes correspond to the local file header signature defined the .ZIP File](#)  
2 [Format Specification from PKWARE, Inc., version 6.2.0 \(2004\)](#)

3 [2. Test that a valid Package Relationships zip item exists](#)

4 [a. As required by §8.3.4 of the Open Packaging Conventions](#)

5 [b. Check that the content type for the package relationships zip item is correctly defined](#)  
6 [in the Content Types stream \(see OPC §8.1.2\) as a relationships part](#)

7 [3. Test that the Package Relationships Part contains a relationship \(see OPC §8.3\) whose](#)  
8 [Target attribute points to a valid FixedDocumentSequence Part](#)

9 [a. As required by §10.2 of this Standard](#)~~XML Paper Specification~~

10 [b. Check that the content type for the FixedDocumentSequence part is correctly defined](#)  
11 [in the Content Types stream as a FixedDocumentSequence part](#)

12 **End of informative text.**



## 1 **E-F. Conformance Requirements**

### 2 **This annex is informative**

3 This annex restates the conformance requirements for producers and consumers implementing  
4 the [Open XML Paper Specification](#). This restatement does not include conformance  
5 requirements from normative references such as ECMA-376:2006.

6 In this annex, conformance requirements are divided into three tables per clause, respectively  
7 containing the conformance requirements that producers and consumers must follow, those  
8 that they should follow, and those that are optional. Each conformance requirement is given a  
9 unique ID comprised of a letter (M – MUST; S – SHOULD; O – OPTIONAL), a requirements  
10 group number, and a unique ID within that clause. Producers and consumers can use these IDs  
11 to report error conditions. If a requirement is removed from this Standard, its ID will not be  
12 reused for any newly added requirement.

### 13 **E-F.1 Implementation Conformance**

#### 14 **E-F.1.1 MUST Conformance Requirements**

15 *Table F-1. Implementation MUST conformance requirements*

ID	Rule	Reference
M0.1	A conforming consumer MUST interpret and process the contents of OpenXPS Document instances in a manner conforming to this Standard.	2.2
M0.2	A conforming consumer is NOT REQUIRED to interpret or process all of the content in an OpenXPS Document instance.	2.2
M0.3	A conforming consumer MUST NOT instantiate an error condition in response to OpenXPS Document content conforming to this Standard.	2.2
M0.4	When "OPTIONAL" or "RECOMMENDED" features contained within OpenXPS Document instances are accessed by a consumer, the consumer MUST interpret and process those features in a manner conforming to this Standard.	2.2
M0.5	Any OpenXPS Document instances a conforming producer creates MUST conform to this Standard.	2.2
M0.6	A conforming producer MUST NOT introduce any non-conforming OpenXPS Document content when modifying an OpenXPS Document instance	2.2
M0.7	When a conforming producer chooses to use an "OPTIONAL" or "RECOMMENDED" feature in an OpenXPS Document instance, then the producer MUST create or modify that feature in a manner conforming to this Standard	2.2

**~~E.1.2~~F.1.2 SHOULD Conformance Requirements**

Table F-2. Implementation SHOULD conformance requirements

ID	Rule	Reference
S0.1	A conformant consumer SHOULD instantiate an error condition when OpenXPS Document content not conforming to this Standard is encountered	2.2

**~~E.2~~F.2 OpenXPS Document Format****~~E.2.1~~F.2.1 MUST Conformance Requirements**

Table F-3. OpenXPS Document format MUST conformance requirements

ID	Rule	Reference
M1.1	OpenXPS Documents MUST observe all conformance requirements of the OPC Standard, except where specifically noted otherwise in this Standard.	8, 9.2
M1.2	The OpenXPS Document format MUST use a ZIP archive for its physical model.	8.2

**~~E.2.2~~F.2.2 SHOULD Conformance Requirements**

Table F-4. OpenXPS Document format SHOULD conformance requirements

ID	Rule	Reference
S1.1	OpenXPS Documents SHOULD observe all recommendations of the OPC Standard, except where indicated otherwise .	8

**~~E.3~~F.3 Parts and Relationships****~~E.3.1~~F.3.1 MUST Conformance Requirements**

Table F-5. Parts and Relationships MUST conformance requirements

ID	Rule	Reference
M2.1	All content to be rendered MUST be contained in the OpenXPS Document.	9.1, 9.1.1,9.1.5,

ID	Rule	Reference
		9.1.7
M2.2	Each part contained in an OpenXPS Document MUST use only the appropriate content type.	9.1, 9.1.7.3
M2.3	An OpenXPS Document MUST contain exactly one FixedDocumentSequence part per fixed payload.	9.1, 9.1.2
M2.4	An OpenXPS Document MUST contain at least one FixedDocument part per fixed payload.	9.1
M2.5	An OpenXPS Document MUST contain at least one FixedPage part per fixed payload.	9.1
M2.6	A <Glyphs> element in FixedPage markup MUST reference a Font part that exists in the OpenXPS Document.	9.1
M2.7	An <ImageBrush> element in FixedPage markup MUST reference an Image part that exists in the OpenXPS Document.	9.1
M2.8	If FixedPage markup references a Remote Resource Dictionary part, it MUST be included in the OpenXPS Document	9.1
M2.9	This requirement was removed prior to Edition 1 of this Standard.	
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, 9.1.5, 9.1.7, 15.2.3
M2.11	This requirement was removed prior to Edition 1 of this Standard.	
M2.12	<p>A Restricted Font relationship is REQUIRED for each print and preview font used, from the FixedDocument part to the preview and print Font part.</p> <p>For preview and print embedding, a producer MUST add a Restricted Font relationship to the FixedDocument part of the document containing the font.</p> <p>If a producer embeds a font with the print and preview restriction bit set, it MUST also add a Restricted Font relationship from the FixedDocument part that includes the FixedPage referencing the font to the restricted font.</p> <p>When editing content, producers MUST instantiate an error condition when encountering any font with the print and preview restriction bit set for which no Restricted Font relationship has been added to the FixedDocument part.</p>	9.1.1, 9.1.7.2, 9.1.7.4
M2.13	Exactly one StartPart relationship is REQUIRED.	9.1.1
M2.14	The StartPart relationship MUST point from the package to the FixedDocumentSequence part that is the primary fixed payload	9.1, 9.1.1

ID	Rule	Reference
	root.	
M2.15	The order of <DocumentReference> elements in a FixedDocumentSequence part MUST be preserved.	9.1.2
M2.16	The order of <PageContent> elements in a FixedDocument MUST be preserved.	9.1.3
M2.17	JPEG image parts MUST contain images that conform to the JPEG specification.	9.1.5.1
M2.18	PNG image parts MUST contain images that conform to the PNG specification.	9.1.5.2
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
M2.26	OpenXPS 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	OpenXPS Document consumers MUST support TIFF images using CCITT bilevel encoding.	9.1.5.3
M2.29	OpenXPS Document consumers MUST support CMYK TIFF images.	9.1.5.3
M2.30	OpenXPS 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	OpenXPS Document consumers MUST support TIFF images using LZW compression.	9.1.5.3
M2.32	OpenXPS Document consumers MUST support TIFF images using differencing predictors.	9.1.5.3
M2.33	OpenXPS Document consumers MUST support TIFF images using JPEG compression (compression mode 6 only).	9.1.5.3
M2.34	OpenXPS Document consumers MUST support TIFF images with an embedded ICC profile.	9.1.5.3
M2.35	<a href="#">JPEG XR image parts MUST conform to the JPEG XR specification.</a> <del>Windows Media Photo image files MUST conform to</del>	9.1.5.4

ID	Rule	Reference
	<del>the Windows Media Photo specification.</del>	
M2.36	Each FixedPage part MUST NOT have more than one thumbnail part attached.	9.1.6
M2.37	Thumbnails MUST be either JPEG or PNG images	9.1.6
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 and $n-1$ inclusive, where $n$ is the number of font faces in the TrueType Collection.	9.1.7
M2.39	OpenXPS Documents MUST support the <a href="#">OpenType font format (ISO/IEC 14496-22:2007)</a> , including TrueType and CFF fonts. <del>Although a subsetted font does not contain all the glyphs in the original font, it MUST be a valid Open Font Format file. 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.</del>	9.1.7, 9.1.7.1
M2.40	<a href="#">Producers MUST observe the guidelines and mechanisms described below in order to honor the licensing rights specified in Open Font Format.</a> <del>Producers MUST honor the licensing rights specified in OpenType fonts by following the embedding and obfuscation mechanisms described in this Standard.</del>	9.1.7.2
M2.41	Consumers MUST be able to process OpenXPS Documents using any combination of the embedding and obfuscation mechanisms described in this Standard (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 OpenXPS Document markup or hierarchical structure from the FixedDocument containing such a font downwards. A producer MUST NOT modify or edit the OpenXPS Document markup or hierarchical structure starting from the <FixedDocument> element. When editing content, producers MUST NOT edit a document where the FixedDocument part has a Restricted Font relationship.	9.1.7.2, 9.1.7.4
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

ID	Rule	Reference
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 be the GUID generated during the font obfuscation process, with or without an extension.	9.1.7.3
M2.55	<u>When processing &lt;Glyphs&gt; elements, the consumer MUST first select a cmap table from the Open Font Format following the order of preference shown below (highest listed first).</u> <del>When processing &lt;Glyphs&gt; 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.</del>	9.1.7.5
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
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
M2.58	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Consumers MUST process all PrintTicket parts when an OpenXPS Document is printed.	

ID	Rule	Reference
M2.59	PrintTicket parts can be attached only to FixedDocumentSequence, FixedDocument, and FixedPage parts, and each of these parts MUST attach no more than one PrintTicket.	9.1.9
M2.60	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Consumers MUST process job-level, document-level and page-level settings of PrintTicket parts associated with FixedDocumentSequence parts.	
M2.61	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M2.62	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M2.63	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When processing a PrintTicket, consumers MUST first remove all levels of PrintTicket content not applicable to the current element.	
M2.64	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When processing a PrintTicket, consumers MUST second validate the PrintTicket according to the methods defined in the PrintTicket Validation Checklist of the Print Schema documentation.	
M2.65	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Following validation of a PrintTicket, the printing consumer MUST properly interpret the print settings according to the rules for merging two PrintTicket parts.	
M2.66	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M2.67	This requirement was removed prior to Edition 1 of this Standard;	

ID	Rule	Reference
	<p>its description is retained here for historical purposes.</p> <p>If there is a print setting merge conflict between different PrintTicket levels, the setting from the most specific ticket MUST take precedence.</p>	
M2.68	<p>Consumers MUST use semantic document structure provided in included DocumentStructure and StoryFragments parts in preference to any other analysis method of generating such structure.</p>	9.1.11
M2.69	<p>Consumers MUST support Markup Compatibility and Extensibility elements and attributes in DocumentStructure, FixedDocument, FixedDocumentSequence, FixedPage, Relationships, Remote Resource Dictionary, SignatureDefinitions, and StoryFragments parts.</p> <p>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 OpenXPS Document markup.</p>	9.3.1, 9.3.2
M2.70	<p>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 Standard), that declaration MUST NOT name any encoding other than UTF-8 or UTF-16.</p>	9.3.2
M2.71	<p>DTD content MUST NOT be used in the XML markup defined in this Standard, and consumers MUST instantiate an error condition when encountering DTD content.</p>	9.3.2
M2.72	<p>XML content MUST be valid against the corresponding W3C XSD schema defined in this Standard. 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.</p>	9.3.2
M2.73	<p>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 Standard.</p>	9.3.2
M2.74	<p>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 instantiate an error condition when encountering properties that are specified in both ways.</p>	9.3.3.2
M2.75	<p>OpenXPS Document markup MUST NOT use the xml:space attribute.</p>	9.3.4
M2.76	<p>The language of the contents of an OpenXPS Document MUST be identified using the xml:lang attribute, the value of which is inherited by child and descendant elements. When the language</p>	9.3.5.1



ID	Rule	Reference
	of the contents is unknown and is required, the value "und" (undetermined) MUST be used.	
M2.77	Producers that generate a relationship MUST include the target part in the OpenXPS 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 instantiate an error condition if the part is not included in the OpenXPS Document.	9.1.1
M2.78	Consumers MUST support JPEG images that contain the EXIF-specified APP1 marker and interpret the EXIF color space correctly.	9.1.5.1
M2.79	OpenXPS 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
M2.80	Each <DocumentReference> element in a FixedDocumentSequence part MUST reference a FixedDocument part by relative URI.	9.1.2
M2.81	Each <PageContent> element in a FixedDocument part MUST reference a FixedPage part by relative URI.	9.1.3
M2.82	<ImageBrush> and <Glyphs> elements MUST reference Image and Font parts by relative URI.	9.1.4
M2.83	If the ExtraSamples tag value is 0, the associated alpha data in this channel MUST be ignored	9.1.5.3
M2.84	The payload <a href="#">containing</a> <del>for</del> an OpenXPS Document may include additional parts not defined by this Standard. Consumers MUST ignore parts in valid OpenXPS Documents that they do not understand.	9.1
M2.85	Consumers MUST ensure that they can distinguish between the uses of those markers listed in Table 9-3 and other data that is recorded using the same markers.	9.1.5.1
M2.86	Signature definitions MUST conform to the Signature Definitions schema as defined in §A.1.	9.1.10
M2.87	The order of child property elements is significant: they MUST occur before any contents of the parent element and they MUST appear in the sequence specified in the schema	9.3.3.2.3
M2.88	xml:lang is REQUIRED for <FixedPage> elements.	9.3.5.1
M2.89	xml:lang MUST NOT be used on any other fixed page markup element [than <FixedPage>, <Canvas>, <Path>, or <Glyphs>].	9.3.5.1
M2.90	xml:lang is REQUIRED for the <DocumentOutline> element for document structure.	9.3.5.1

ID	Rule	Reference
<a href="#">M2.91</a>	<a href="#">JPEG XR image parts MUST use the Tag-based file format defined in Annex A of the JPEG XR specification.</a>	9.1.5.4

1 **~~E.3.2~~F.3.2 SHOULD Conformance Requirements**

2 *Table F-6. Parts and Relationships SHOULD conformance requirements*

ID	Rule	Reference
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
S2.2	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Each <DocumentReference> element in a FixedDocumentSequence part SHOULD reference a FixedDocument part by relative URI.	
S2.3	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Each <PageContent> element in a FixedDocument part SHOULD reference a FixedPage part by relative URI.	
S2.4	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. <ImageBrush> and <Glyphs> elements SHOULD reference Image and Font parts by relative URI.	
S2.5	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. (See [M8.41] and [M8.42].) Color profiles embedded in image files SHOULD be used if present and compatible with this Standard.	
S2.6	It is RECOMMENDED that JPEG image part names end with the extension ".jpg".	9.1.5.1
S2.7	The use of CMYK JPEG images is NOT RECOMMENDED. TIFF or <del>Windows Media Photo</del> JPEG XR images SHOULD be used instead to represent CMYK images.	9.1.5.1, 0, 15.3.7
S2.8	It is RECOMMENDED that PNG image part names end with the extension ".png".	9.1.5.2
S2.9	It is RECOMMENDED that TIFF image part names end with	9.1.5.3

ID	Rule	Reference
	the extension ".tif".	
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
S2.11	Given the wide variety of incompliant TIFF images in 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.	9.1.5.3
S2.12	<u>It is RECOMMENDED that JPEG XR image part names end with the extension ".jxr".</u> <del>It is RECOMMENDED that Windows Media Photo images end with the extension ".wdp".</del>	9.1.5.4
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
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
S2.15	Producers SHOULD use Unicode-encoded fonts.	9.1.7, 9.1.7.5
S2.16	For fonts with "Installable embedding" licensing intent, producers SHOULD perform embedded font obfuscation.	9.1.7.2
S2.17	For fonts with "Installable embedding" licensing intent, consumers SHOULD NOT extract or permanently install the font.	9.1.7.2
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
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 OpenXPS Documents. Instead, producers SHOULD always perform font obfuscation, and consumers SHOULD never extract or permanently install fonts.	9.1.7.3
S2.20	It is RECOMMENDED that the extension of an obfuscated Font part name be ".odttf" for TrueType fonts and ".odttc" for TrueType collections.	9.1.7.3

ID	Rule	Reference
S2.21	<p>This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.</p> <p>Producers SHOULD include only PrintTicket settings that support portability of the OpenXPS Document.</p>	
S2.22	<p>This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.</p> <p>Producers SHOULD only attach PrintTicket parts containing only document-level and page-level settings with FixedDocument parts.</p>	
S2.23	<p>This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.</p> <p>Producers SHOULD only attach PrintTicket parts containing only page-level settings with FixedPage parts.</p>	
S2.24	<p>The FixedDocumentSequence part SHOULD follow the part name recommendation <code>"&lt;FixedDocSeq&gt;.fdseq"</code> where <code>&lt;FixedDocSeq&gt;</code> is the name of the FixedDocumentSequence. The FixedDocumentSequence SHOULD use the extension <code>".fdseq"</code>.</p>	9.2
S2.25	<p>A FixedDocument part SHOULD follow the part name recommendation <code>"&lt;Documents/&lt;n&gt;/&lt;FixedDocument&gt;.fdoc"</code> where <code>&lt;n&gt;</code> is a numeral that indicates the ordinal position of the fixed document in the fixed document sequence and <code>&lt;FixedDocument&gt;</code> is the name of the fixed document. FixedDocument parts SHOULD use the extension <code>".fdoc"</code>.</p>	9.2
S2.26	<p>A FixedPage part SHOULD follow the part name recommendation <code>"&lt;Documents/&lt;n&gt;/Pages/&lt;m&gt;.fpage"</code> where <code>&lt;n&gt;</code> represents the document that includes this page and <code>&lt;m&gt;</code> is the page number. FixedPage parts SHOULD use the extension <code>".fpage"</code>.</p>	9.2
S2.27	<p>A resource that is specific to a particular document SHOULD follow the part name recommendation <code>"&lt;Documents/&lt;n&gt;/Resources/&lt;Resource&gt;"</code> where <code>&lt;n&gt;</code> is the document that uses the resource and <code>&lt;Resource&gt;</code> is the segments identifying the particular resource.</p> <p>A resource that is shared across multiple documents SHOULD follow the part name recommendation <code>"&lt;Resources/&lt;Resource&gt;"</code>.</p> <p>A Font resource SHOULD use <code>"Fonts/&lt;FontName&gt;.&lt;FontExt&gt;"</code> for its <code>&lt;Resource&gt;</code> value, where <code>&lt;FontExt&gt;</code> SHOULD be either <code>".tff"</code> or <code>".odtff"</code> for</p>	9.2

ID	Rule	Reference
	<p>non-obfuscated and obfuscated fonts respectively.</p> <p>An Image resource SHOULD use "Images/&lt;ImageName&gt;.&lt;ImageExt&gt;" for its &lt;Resource&gt; value, where &lt;ImageExt&gt; is the correct extension for the image type.</p> <p>A Remote Resource Dictionary resource SHOULD use "Dictionaries/&lt;DictName&gt;.dict" for its &lt;Resource&gt; value. A Remote Resource Dictionary SHOULD use ".dict" as its extension.</p> <p>&lt;FontName&gt;, &lt;ImageName&gt;, and &lt;DictName&gt; SHOULD be a string representation of a GUID value if the resource is a shared resource.</p>	
S2.28	<p>A DocumentStructure part SHOULD follow the part name recommendation "/Documents/&lt;n&gt;/Structure/&lt;DocStruct&gt;.struct" where &lt;n&gt; is the fixed document that the document structure applies to. DocumentStructure parts SHOULD use the extension ".struct".</p>	9.2
S2.29	<p>A StoryFragments part SHOULD follow the part name recommendation "/Documents/&lt;n&gt;/Structure/Fragments/&lt;m&gt;.frag" where &lt;n&gt; is the fixed document that contains the story fragments and &lt;m&gt; is the page number the StoryFragments part applies to. StoryFragments parts SHOULD use the extension ".frag".</p>	9.2
S2.30	<p>ICC profile parts SHOULD follow the part name recommendation "/Documents/&lt;n&gt;/Metadata/&lt;ProfileName&gt;.&lt;ProfileExt&gt;" where &lt;n&gt; is the fixed document that contains the profile.</p> <p>ICC profile parts shared across multiple documents SHOULD follow the part name recommendation "/Metadata/&lt;ProfileName&gt;.&lt;ProfileExt&gt;". In this case, &lt;ProfileName&gt; SHOULD be a string representation of a GUID value.</p> <p>The &lt;ProfileExt&gt; SHOULD be appropriate to the color profile type, such as ".icm".</p>	9.2
S2.31	<p>Thumbnail parts SHOULD follow the part name recommendation "/Documents/&lt;n&gt;/Metadata/&lt;ThumbName&gt;.&lt;ThumbExt&gt;" where &lt;n&gt; is the fixed document that contains the thumbnail.</p> <p>If the Thumbnail part represents the package as a whole, it SHOULD follow the part name recommendation "/Metadata/&lt;ThumbName&gt;.&lt;ThumbExt&gt;". In this case, &lt;ThumbName&gt; SHOULD be a string representation of a</p>	9.2

ID	Rule	Reference
	<p>GUID value.</p> <p>The <i>&lt;ThumbExt&gt;</i> SHOULD be appropriate to the image type, either ".png" or ".jpg".</p>	
S2.32	<p>PrintTicket part names associated with the entire job SHOULD be associated via relationship with the FixedDocumentSequence part and contain two segments, using "/Metadata/" as the first segment.</p> <p>PrintTicket parts associated with a particular fixed document or fixed page SHOULD contain four segments, using "/Documents/<i>n</i>/Metadata/" as the first three segments, where <i>&lt;n&gt;</i> is the fixed document that uses these parts.</p> <p>PrintTicket parts based on XML SHOULD use the extension ".xml".</p>	9.2
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/<i>&lt;n&gt;</i>/Other/<i>&lt;PartName&gt;</i>", where <i>&lt;n&gt;</i> is the fixed document to which the part belongs.</p>	9.2
S2.34	<p>Consumers SHOULD support JPEG images that contain JFIF-specified APP0 and ICC-specified APP2 markers.</p>	9.1.5.1
S2.35	<p>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 instantiate an error condition.</p>	9.1.7
S2.36	<p>If the consumer understands the content of the PrintTicket, then the PrintTicket part SHOULD be processed when the OpenXPS Document is printed.</p>	9.1.9
S2.37	<p>For images that have a constant opacity, producers SHOULD NOT use the image format alpha channel; the Opacity attribute in the <i>&lt;ImageBrush&gt;</i> element SHOULD be used instead.</p>	9.1.5

1 **E-3.3F.3.3 OPTIONAL Conformance Requirements**

2 *Table F-7. Parts and Relationships OPTIONAL conformance requirements*

ID	Rule	Reference
O2.1	Thumbnail parts MAY be included in an OpenXPS Document	9.1
O2.2	PrintTicket parts MAY be included in an OpenXPS Document.	9.1
O2.3	ICC Profile parts MAY be included in an OpenXPS Document.	9.1

O2.4	DocumentStructure parts MAY be included in an OpenXPS Document.	9.1
O2.5	StoryFragments parts MAY be included in an OpenXPS Document.	9.1
O2.6	SignatureDefinitions parts MAY be included in an OpenXPS Document.	9.1, 9.1.10
O2.7	DiscardControl parts MAY be included in an OpenXPS Document.	9.1
O2.8	A Core Properties relationship MAY be included in an OpenXPS Document, from the package to the Core Properties part.	9.1.1
O2.9	A Digital Signatures Origin relationship MAY be included in an OpenXPS Document, from the package to the Digital Signature Origin part.	9.1.1
O2.10	Digital Signature relationships MAY be included in an OpenXPS Document, from the Digital Signature Origin part to a Digital Signature XML Signature part.	9.1.1
O2.11	Digital Signature Certificate relationships MAY be included in an OpenXPS Document, from a Digital Signature XML Signature part to the Digital Signature Certificate part.	9.1.1
O2.12	Digital Signature Definitions parts MAY be included in an OpenXPS Document, from a FixedDocument part to the Digital Signature Definitions part.	9.1.1
O2.13	DiscardControl relationships MAY be included in an OpenXPS Document, from the package to a DiscardControl part.	9.1.1
O2.14	DocumentStructure relationships MAY be included in an OpenXPS Document, from a FixedDocument part to the DocumentStructure part.	9.1.1
O2.15	PrintTicket relationships MAY be included in an OpenXPS Document, from a FixedDocumentSequence, FixedDocument, or FixedPage part to a PrintTicket part.	9.1.1
O2.16	StoryFragments relationships MAY be included in an OpenXPS Document, from a FixedPage part to a StoryFragments part.	9.1.1
O2.17	Thumbnail relationships MAY be included in an OpenXPS 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 OpenXPS 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 OpenXPS Document MAY also sign it against a signature definition.	9.1.10
O2.26	This requirement was removed prior to Edition 1 of this Standard.	
O2.27	Consumers MAY provide an algorithmic construction of the structure of an OpenXPS 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 OpenXPS 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 OpenXPS 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 OpenXPS 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 Standard.	9.3.4
O2.33	xml:lang MAY be used with <Canvas>, <Path>, and <Glyphs> elements.	9.3.5.1
O2.34	xml:lang is OPTIONAL for the <OutlineEntry> element.	9.3.5.1

## **E.4F.4 Documents**

### **E.4.F.4.1 MUST Conformance Requirements**

Table F-8. Document MUST conformance requirements

ID	Rule	Reference
M3.1	The order of <DocumentReference> elements MUST match the order of the documents in the fixed document sequence.	10.1
M3.2	The Source attribute of the <DocumentReference> element MUST specify a FixedDocument part within the OpenXPS Document.	10.1.1
M3.3	Producers MUST NOT produce a document with multiple <DocumentReference> elements that reference the same fixed document.	10.1.1
M3.4	The order of <PageContent> elements MUST match the order of the pages in the document.	10.2
M3.5	The Source attribute of the <PageContent> element MUST specify a FixedPage part within the OpenXPS Document.	10.2.1
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 OpenXPS Document, even in other fixed documents within the fixed payload.	10.2.1
M3.7	If the attribute is specified, the BleedBox BleedOriginX value MUST be less than or equal to 0.	10.3.1
M3.8	If the attribute is specified, the BleedBox BleedOriginY value MUST be less than	10.3.1



	or equal to 0.	
M3.9	If the attribute is specified, the <code>BleedBox BleedWidth</code> value MUST be greater than or equal to the <code>Width</code> attribute value plus the absolute value of the <code>BleedBox BleedOriginX</code> value.	10.3.1
M3.10	If the attribute is specified, the <code>BleedBox BleedHeight</code> value MUST be greater than or equal to the fixed page <code>Height</code> value plus the absolute value of the <code>BleedBox BleedOriginY</code> value.	10.3.1
M3.11	If the attribute is specified, the <code>ContentBox ContentOriginX</code> value MUST be greater than or equal to 0 and less than the fixed page <code>Width</code> attribute value	10.3.2
M3.12	If the attribute is specified, the <code>ContentBox ContentOriginY</code> value MUST be greater than or equal to 0 and less than the fixed page <code>Height</code> attribute value.	10.3.2
M3.13	If the attribute is specified, the <code>ContentBox ContentWidth</code> value MUST be less than or equal to the difference between the fixed page <code>Width</code> attribute value and the <code>ContentBox ContentOriginX</code> value.	10.3.2
M3.14	If the attribute is specified, the <code>ContentBox ContentHeight</code> value MUST be less than or equal to the difference between the fixed page <code>Height</code> attribute value and the <code>ContentBox ContentOriginY</code> value.	10.3.2
M3.15	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When rendering a fixed page for printing, consumers MUST be aware of the interaction between the fixed page markup and the <code>PrintTicket</code> settings.	
M3.16	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When the <code>PrintTicket</code> specifies the page scaling option <code>FitApplicationBleedSizeToPageImageableSize</code> , printing consumers MUST scale the bleed box (producer bleed size) to the <code>PageImageableSize</code> , preserving the aspect ratio.	
M3.17	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When the <code>PrintTicket</code> specifies the page scaling option <code>FitApplicationContentSizeToPageImageableSize</code> , printing consumers MUST scale the content box (producer content size) to the <code>PageImageableSize</code> , preserving the aspect ratio.	
M3.18	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When the <code>PrintTicket</code> specifies the page scaling option <code>FitApplicationMediaSizeToPageImageableSize</code> , printing consumers MUST scale the height and width (producer media size) to the <code>PageImageableSize</code> , preserving the aspect ratio.	
M3.19	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. When the <code>PrintTicket</code> specifies the page scaling option <code>FitApplicationMediaSizeToPageMediaSize</code> , printing consumers MUST scale the height and width (producer media size) to the <code>PageMediaSize</code> , preserving the aspect ratio.	
M3.20	The <code>x:Key</code> attribute of the <code>&lt;Canvas&gt;</code> element MUST be present when the element is defined in a resource dictionary. It MUST NOT be specified outside a	10.4

	resource dictionary.	
M3.21	The <PageContent> element has one allowable child element, <PageContent.LinkTargets>, and it MUST NOT contain more than a single child element.	10.2.1
M3.22	The fixed page MUST specify a height, width, and default language.	10.3

### 1 **E-4.2F.4.2 SHOULD Conformance Requirements**

2 *Table F-9. Document SHOULD conformance requirements*

ID	Rule	Reference
S3.1	Specifying a ContentBox attribute for the <FixedPage> element is RECOMMENDED.	10.3, 10.3.2
S3.2	Invalid bleed box specifications SHOULD be ignored in favor of the default bleed box.	10.3.1
S3.3	Invalid content box specifications SHOULD be ignored in favor of the default content box.	10.3.2
S3.4	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.  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.	
S3.5	By default, consumers SHOULD clip to the FixedPage Width and Height.	10.3.3

### 3 **E-4.3F.4.3 OPTIONAL Conformance Requirements**

4 *Table F-10. Document OPTIONAL conformance requirements*

ID	Rule	Reference
O3.1	The positioning, scaling, orientation, and clipping of FixedPage content when mapping to physical media MAY be controlled by settings provided in the PrintTicket.	10.3.3
O3.2	Consumers MAY provide implementation-defined mechanisms to select alternative clipping strategies.	10.3.3

1 **E-5F.5 Graphics**2 **E-5.1F.5.1 MUST Conformance Requirements**3 *Table F-11. Graphics MUST conformance requirements*

ID	Rule	Reference
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
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
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
M4.4	The <PathGeometry> element's Figures attribute can be used to describe the path figures the geometry contains using abbreviated syntax with the exception that the FillRule command MUST NOT be used.	11.1.3
M4.5	The x or y radius in the Size attribute MUST NOT be negative.	11.2.2.2.1
M4.6	This [FillRule] command MUST appear only as the first command in the abbreviated geometry syntax.	11.2.3
M4.7	This [FillRule] command MUST NOT be specified in the value of the Figures attribute of the <PathGeometry> element.	11.2.3
M4.8	The first figure in a geometry MUST begin with a Move command.	11.2.3

4 **E-5.2F.5.2 SHOULD Conformance Requirements**5 *Table F-12. Graphics SHOULD conformance requirements*

ID	Rule	Reference
S4.1	Line segments and curve segments SHOULD NOT be specified as zero-length.	11.2.2.1

**~~E.5.3~~F.5.3 OPTIONAL Conformance Requirements**

Table F-13. Graphics OPTIONAL conformance requirements

ID	Rule	Reference
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.	11.1, 19.39
O4.2	A path geometry MAY define the fill algorithm to be used on the component path figures.	11.2
O4.3	Abbreviated geometry syntax MAY be used to specify a geometry of one or more figures comprised of multiple segments.	11.2.3
O4.4	If entering more than one drawing command of the same type sequentially, the duplicate command entry MAY be omitted.	11.2.3
O4.5	Every geometry MAY specify one or more figures, and MAY be preceded by a FillRule command where allowed.	11.2.3
O4.6	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.	11.2.3

**~~E.6~~F.6 Text****~~E.6.1~~F.6.1 MUST Conformance Requirements**

Table F-14. Text MUST conformance requirements

ID	Rule	Reference
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
M5.2	If the UnicodeString attribute of the <Glyphs> element <u>is not specified or contains specifies</u> an empty <u>string value</u> (“” or “{}”), <u>and if</u> the Indices attribute is not specified or <u>contains &lt;nothing&gt; is empty</u> , <u>then</u> a consumer MUST instantiate an error condition.	12.1, 12.1.4
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
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 (“” or “{}”). If a ClusterMapping is not specified within	12.1.3

	a GlyphMapping entry, the code unit count is 1.	
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
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
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
M5.8	This requirement was removed prior to Edition 1 of this Standard.	
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
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 control marks are mapped in the CMAP table.	12.1.4
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
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
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
M5.14	Producers MUST lay out algorithmically italicized glyphs using exactly the same advance widths as when not algorithmically italicized.	12.1.5
M5.15	Producers MUST NOT specify text that is both right-to-left (BidiLevel attribute set to an odd value) and vertical (IsSideways attribute set to true).	12.1.6.2
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
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
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
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	12.1.7

	font name and MUST render the embedded version of the font.	
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
M5.21	This requirement was removed prior to Edition 1 of this Standard.	
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
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
M5.24	If the Indices attribute specifies a GlyphIndex that does not exist in the font, the consumer MUST instantiate an error condition.	12.1.3
M5.25	The Indices attribute MUST adhere to the glyph specification syntax.	12.1.3
M5.26	AdvanceWidth's advance MUST be 0 or greater.	12.1.3
M5.27	For larger blocks of text, the producer MAY specify the xml:lang attribute on the <Canvas> element.	12.1.8

## 1 ~~E-6.2~~[F.6.2](#) **SHOULD Conformance Requirements**

### 2 *Table F-15. Text SHOULD conformance requirements*

<b>ID</b>	<b>Rule</b>	<b>Reference</b>
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
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
S5.3	If alternate vertical character representations are available in the font, the producer SHOULD use those in preference to the lsSideways attribute and provide their glyph indices in the Indices attribute.	12.1.6
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
S5.5	Specifying a UnicodeString for <Glyphs> elements is RECOMMENDED, as it supports searching, selection, and accessibility.	12.1.4
<a href="#">S5.6</a>	<a href="#">When rendering glyphs where the StyleSimulations value is specified as BoldSimulation, consumers SHOULD offset each glyph up and to the right by 1% of the em size so that baseline and left edge alignments are preserved.</a>	12.1.5

1 **E-6-3F.6.3** **OPTIONAL Conformance Requirements**2 *Table F-16. Text OPTIONAL conformance requirements*

ID	Rule	Reference
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 Standard.	12.1.4
O5.2	Producers MAY choose to generate UnicodeString attribute values that are not normalized by any Unicode-defined algorithm.	12.1.4
O5.3	Consumers that understand the device font name MAY ignore the embedded font and use the device-resident version.	12.1.7
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
O5.5	Glyph advance widths MAY be omitted from markup where the advance width desired is specified in the font tables, once adjusted for algorithmic boldening.	12.1.10.2
O5.6	Glyph horizontal and vertical offsets MAY be omitted from markup where the offset is 0.0.	12.1.10.2
O5.7	The <Glyphs> element MAY have an Indices attribute.	12.1.3
O5.8	The glyph specifications within the Indices attribute are OPTIONAL.	12.1.3
O5.9	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.	12.1.3
O5.10	The Indices attribute MAY include glyph placement information.	12.1.3
O5.11	The GlyphIndex entry MAY be empty.	12.1.3
O5.12	A cluster map specification MAY precede the glyph specification for the first glyph of the cluster.	12.1.3.1
O5.13	The language defaults to the value specified for the xml:lang attribute of the <FixedPage> element but MAY be overridden by an xml:lang attribute on a <Glyphs> element.	12.1.8

## ~~E.7~~F.7 Brushes

### ~~E.7.1~~F.7.1 MUST Conformance Requirements

Table F-17. Brushes MUST conformance requirements

ID	Rule	Reference
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
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
M6.3	An <ImageBrush> element MUST reference a JPEG, PNG, TIFF, or <del>Windows Media Photo</del> <u>JPEG XR</u> Image part within the OpenXPS Document package.	13.2
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
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
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
M6.7	ViewboxUnits specifies the unit type for the Viewbox attribute. MUST have the value "Absolute".	13.4
M6.8	ViewportUnits specifies the unit type for the Viewport attribute. MUST have the value "Absolute".	13.4

## ~~E.8~~F.8 Common Properties

### ~~E.8.1~~F.8.1 MUST Conformance Requirements

Table F-18. Common properties MUST conformance requirements

ID	Rule	Reference
M7.1	Individual resource values MUST be specified within a resource dictionary.	14.2
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
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



M7.4	A remote resource dictionary MUST follow the requirements that apply to inline resource dictionaries.	14.2.3.1
M7.5	A remote resource dictionary MUST NOT contain any resource definition children that reference another remote resource dictionary.	14.2.3.1
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
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
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
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
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
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
M7.12	The Opacity property attribute value MUST fall within the 0 (fully transparent) to 1 (fully opaque) range, inclusive.	14.1
M7.13	The <Canvas.Resources> or <FixedPage.Resources> property elements MUST precede any property elements of the <Canvas> or <FixedPage> elements.	14.2
M7.14	The <Canvas.Resources> or <FixedPage.Resources> property elements MUST precede any path, glyphs, or canvas children of the <Canvas> or <FixedPage> elements.	14.2
M7.15	<FixedPage.Resources> and <Canvas.Resources> elements that include a remote resource dictionary MUST include exactly one <ResourceDictionary> element.	14.2.3.1
M7.16	The value of the x:Key attribute MUST be unique within the resource dictionary.	14.2.5

## 1 [F.8.2 SHOULD Conformance Requirements](#)

### 2 [Table F-19. Common properties SHOULD conformance requirements](#)

<a href="#">ID</a>	<a href="#">Rule</a>	<a href="#">Reference</a>
<a href="#">S7.1</a>	<a href="#">A consumer SHOULD instantiate an error condition if a static resource reference cannot be resolved, or if it <i>can</i> be resolved but the resource type does not match the usage at the location of reference.</a>	14.2.4

[S7.2](#) [A consumer SHOULD instantiate an error condition occurs if the search has continued to the root <FixedPage> element and a specified resource has not been found](#) 14.2.5

1 ~~E.8.2~~[F.8.3](#) **OPTIONAL Conformance Requirements**

2 *Table F-20. Common properties OPTIONAL conformance requirements*

ID	Rule	Reference
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
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
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
O7.4	This requirement was removed prior to Edition 1 of this Standard.	
O7.5	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
O7.6	A resource definition MAY reference a previously defined resource with the same name that is defined in an ancestor resource dictionary.	14.2.5
O7.7	An abbreviated matrix transformation syntax MAY be used to specify a RenderTransform or Transform attribute value.	14.4

3 ~~E.9~~[F.9](#) **Color**

4 ~~E.9.1~~[F.9.1](#) **MUST Conformance Requirements**

5 *Table F-21. Color MUST conformance requirements*

ID	Rule	Reference
M8.1	Consumers MUST support alpha and gradient blending in sRGB.	15.1
M8.2	Consumers MUST support sRGB colors in image data, using the JPEG, PNG, TIFF, or <del>Windows Media Photo</del> <a href="#">JPEG XR</a> image formats.	15.1
M8.3	Consumers MUST support scRGB color specification in vector data, with and without alpha.	15.1
M8.4	Consumers MUST support scRGB colors in image data, using the <del>Windows Media Photo</del> <a href="#">JPEG XR</a> image format.	15.1

M8.5	Consumers MUST support CMYK colors in vector data.	15.1
M8.6	Consumers MUST support CMYK colors in image data, using the TIFF or <del>Windows Media Photo</del> JPEG XR image formats.	15.1
M8.7	Consumers MUST support N-Channel colors in vector data.	15.1
M8.8	Consumers MUST support N-Channel colors in image data, using the <del>Windows Media Photo</del> JPEG XR image format.	15.1
M8.9	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. It was subsumed by [M8.53]. Consumers MUST support profiles as specified in the ICC specification.	
M8.10	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Consumers MUST support profiles compliant with ICC.1:2001-04 with a Windows Color System (WCS) profile embedded as a private tag.	
M8.11	This requirement was removed prior to Edition 1 of this Standard, and replaced with S8.21; its description is retained here for historical purposes. 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 or gradient blending.	
M8.12	OpenXPS 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, 15.6
M8.13	All ICC profiles used in OpenXPS Documents MUST be an Input profile, an Output profile, a Monitor (RGB) profile, a ColorSpace Conversion profile, or a Named Color profile.	15.1.8
M8.14	Real numbers specified for color channel values of scRGB and ContextColor colors MUST NOT use exponent forms of numbers.	15.2
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
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
M8.17	For N-Channel colors, the context color MUST specify the number of channel float values equal to the number of channels in the profile.	15.2.3, 15.2.5
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.	15.2.5
M8.19	In the case of a named color with an associated tint LUT implemented in an ICC monochrome profile, the profile MUST include an AtoB1Tag (relative colorimetric rendering intent), mapping the named color tint values to valid PCS values.	15.2.6
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.	15.2.6

M8.21	This requirement was removed prior to Edition 1 of this Standard.	
M8.22	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M8.23	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M8.24	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
M8.25	For gradients, the specified blending color space in the blending color space 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 blending color space PrintTicket setting, the color interpolation mode of the gradient brush MUST be used instead.	15.5
M8.26	This requirement was removed prior to Edition 1 of this Standard.	
M8.27	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. If the PageDeviceColorSpaceUsage is set to MatchToDeviceDefault and the profile specified by the PageDeviceColorSpaceURI PrintTicket 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	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. If the PageBlendColorSpace PrintTicket setting is set to ICCProfile, the profile MUST be an output profile, otherwise it MUST be ignored.	
M8.29	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Elements using the named color identified by the DocumentImpositionColor PrintTicket setting MUST appear on all color separations.	
M8.30	If no usable profile is present with an image, then a consumer MUST apply a color rule based on the pixel format. Each pixel format is interpreted to be the encoding of a particular color space as shown in Table 15-2.	9.1.5.1, 9.1.5.2, 9.1.5.3, 9.1.5.4
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. Before the value is used as input for an ICC profile color transformation, it MUST be linearly scaled (with specified	15.2.3, 15.2.4, 15.2.5, 15.2.6

	rounding/clipping) 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.	
M8.32	For 1-channel color, i.e., monochrome, use a monochrome input (or output) profile. The profile MUST include the ICC-optional AToB1Tag (relative colorimetric intent) if the single color is chromatic (not neutral).	15.2.5
M8.33	A named color with an associated tint LUT MUST be implemented in an OpenXPS Document using an associated ICC monochrome profile.	15.2.6
M8.34	The ICC profile for a named color with an associated tint LUT MUST contain the tint LUT for a single named color.	15.2.6
M8.35	The ICC profile for a named color with an associated tint LUT MUST be an ICC monochrome input or output profile.	15.2.6
M8.36	In the case of a named color with an associated tint LUT the ASCII prefix-root-suffix name of the named color MUST be encoded into the profileDescriptionTag of the ICC profile.	15.2.6
M8.37	In the case of a named color with an associated tint LUT the profile header color space signature MUST be 'GRAY'.	15.2.6
M8.38	Two or more named colors implemented in an OpenXPS Document using a single associated profile MUST use an ICC Named Color type profile.	15.2.6
M8.39	An ICC Named Color type profile MUST contain the namedColor2Tag including the ASCII prefix-root-suffix name for each named color.	15.2.6
M8.40	The namedColor2Tag in a Named Color type profile MUST be populated with the ICC PCS color value for each named color.	15.2.6
M8.41	If present and usable, an associated profile MUST be used by consumers.	15.3.8
M8.42	If present and usable, a color profile embedded in an image file MUST be used by consumers when no usable associated profile is present with the image.	15.3.8
M8.43	A producer MUST associate or embed a usable color profile if the color rules of Table 15–2 do not guarantee appropriate color interpretation for an image.	15.3.8
M8.44	Profiles associated as described in Table 15–1, and determined to be usable, MUST be used by consumers.	15.2
M8.45	If no usable profile is present in a context color syntax, then a consumer MUST apply a color rule based on the context color syntax.	15.2
M8.46	Single component integer default for vector data MUST be grayscale with the sRGB non-linearity, black point, and white point.	15.2
M8.47	Three component integer default for vector data MUST be sRGB.	15.2
M8.48	Three component float default for vector data MUST be scRGB.	15.2
M8.49	The specific CMYK to be used as the four component data default for vector data MUST be determined by the consumer.	15.2
M8.50	N-Channel data with $N \leq 3$ and any named color data: the data of the first channel MUST be interpreted independently as grayscale. Other channels are disregarded.	15.2
M8.51	N-Channel with $N > 4$ MUST be treated as four component data using the four component data default for vector data determined by the consumer.	15.2

M8.52	A producer MUST associate or embed a usable color profile if the color rules above do not guarantee appropriate color interpretation for the vector color content.	15.2
M8.53	OpenXPS consumers MUST use associated and embedded ICC profiles, according to the precedence order of §15.3.8 for raster images and according to §15.2 for vector content.	15.1.8
M8.54	Implementations MUST ignore and preserve private tags that they do not understand	15.1.8
M8.55	For Named colors, the OpenXPS context color syntax MUST specify the matching number of tint float values.	15.2.6
M8.56	Consumers MUST support grayscale colors (single channel) in vector data, with and without alpha.	15.1
M8.57	Consumers MUST support grayscale colors in image data, using the JPEG, PNG, TIFF, or HDPhoto image formats.	15.1
M8.58	Producers MUST restrict associated 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
M8.59	If a Producer includes an image with an embedded profile conforming to the requirements of ISO 15076-1, then the Producer MUST associate an ICC profile conforming to the requirements of the older ICC Color Profile specification, ICC.1:2001-04, to have precedence over such an embedded profile, when consumer support of the newer ISO version cannot be ascertained.	15.1.8

1 **~~E-9.2~~F.9.2 SHOULD Conformance Requirements**

2 *Table F-22. Color SHOULD conformance requirements*

ID	Rule	Reference
S8.1	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. It was subsumed by [M8.53] and [M8.41] through [M8.44]. 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.	
S8.2	If consistency of appearance of grayscale images is important, the producer SHOULD adjust the gray tone response curve of such images before adding to the OpenXPS Document.	15.1.8
S8.3	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. A producer of OpenXPS 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).	
S8.4	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
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.	15.2.6
S8.6	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Support for JPEG CMYK images varies by implementation and SHOULD NOT be used in OpenXPS Documents.	
S8.7	For consumers that do perform separation, the occurrence of the document registration named color in a color syntax is <i>only</i> an indicator that the tint level supplied in the syntax SHOULD be used when drawing the registration marking in each colorant separation.	15.4
S8.8	Producers SHOULD create the profile for the document registration named 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
S8.9	A page-level PrintTicket setting can be used to specify the blending color space that SHOULD be used for blending gradients and transparencies. If a consumer understands the blending color space PrintTicket setting, it SHOULD convert all color to the specified blending color space before performing a blend operation.	15.5
S8.10	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
S8.11	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. If the PageDeviceColorSpaceUsage PrintTicket setting is set to OverrideDeviceDefault and the profile specified by the PageDeviceColorSpaceProfileURI PrintTicet 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.	
S8.12	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
S8.13	This requirement was removed prior to Edition 1 of this Standard; its	

	description is retained here for historical purposes. The PageICMRenderingIntent PrintTicket setting SHOULD be ignored for elements using a profile that specifies the rendering intent in the profile.	
S8.14	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. It was subsumed by new rules for named colors in §15.2.3. 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).	
S8.15	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. It was replaced by new rules M8.58 and M8.59. 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.	
S8.16	If the OpenXPS system environment allows the use of ICC ISO 15076-1 profiles, the optional colorantTableTag SHOULD be included in such ISO 15076-1 profiles to indicate the names and corresponding PCS values of the individual N-color colorants.	15.2.5, 15.3.5
S8.17	A profile associated or embedded with an image SHOULD be considered unusable by a consumer if 1) The profile is not compatible with the pixel format of the image, 2) The profile contains optional tags that ambiguate OpenXPS use, and 3) The profile contains invalid tag type signatures that invalidate OpenXPS use.	15.3.8
S8.18	A profile associated as in Table 15–1 SHOULD be considered unusable by a consumer if 1) The profile is not compatible with the context color syntax, 2) The profile contains optional tags that ambiguate OpenXPS use, and 3) The profile contains invalid tag type signatures that invalidate OpenXPS use.	15.2
S8.19	The specific CMYK to be used as the four-component raster data default, and the N-Channel (N=>4) default, is implementation-defined. In the absence of specific requirements the use of CGATS/SWOP TR003 2007 CMYK is recommended.	15.3.8
S8.20	In the absence of ICC rendering intents, in a typical case, with ICC profiles conforming to the ICC Color Profile specification, ICC.1:2001-04, a consumer SHOULD apply the defaults shown in Table 15–3 <del>Table 15.4</del> .	15.6
S8.21	If a consumer recognizes that a profile given in the syntax for a page element matches the page level PrintTicket output-ready ICC profile and that the page level PrintTicket output-ready ICC profile is suitable for the output device conditions, then the consumer SHOULD elect to treat the element colors as output-ready colors and not color-manage them, unless forced to do so for transparency effects or gradient blending	15.1.7
S8.22	The name of the document registration named color is given in the profile's	15.4



	profileDescriptionTag. Such a document registration named color SHOULD be unique for that use in the OpenXPS Document instance	
--	--	--

### 1 ~~E.9.3~~F.9.3 **OPTIONAL Conformance Requirements**

2 *Table F-23. Color OPTIONAL conformance requirements*

ID	Rule	Reference
O8.1	Consumers are not required to handle all color spaces natively through every processing stage, but, rather, MAY convert data specified in a color space other than sRGB to sRGB at an early stage (possibly resulting in reduced fidelity).	15.1
O8.2	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. An ICC profile MAY contain the private tag, "MS00", which specifies an embedded Windows Color System (WCS) profile.	
O8.3	When a named color is used in a gradient brush or with transparency, the result produced by consumers determining the color from the ASCII color name found in the associated ICC Profile MAY differ significantly from the result produced by consumers using the encoded color value of the named color from the associated ICC profile.	15.2.6
O8.4	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. A named color profile MAY be used with images for spot coloring.	
O8.5	Producers MAY elect to generate content that provides registration marks for consumers that perform color separation.	15.4
O8.6	Consumers MAY support alpha and gradient blending with color spaces such as scRGB 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
O8.7	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. If the PageColorManagement PrintTicket setting specifies a value of Driver, the driver MAY color manage elements or convert them to different color spaces.	
O8.8	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
O8.9	OpenXPS producers and consumers MAY provide color management using ICC profiles conforming to the requirements of ISO 15076-1.	15.1.8, 15.6
O8.10	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. The specific CMYK to be used as the four-component raster data default,	

	and the N-Channel (N=>4) default, is implementation-defined. In the absence of specific requirements the use of CGATS/SWOP TR003 2007 CMYK is recommended. Alternatively, a consumer MAY choose to instantiate an error condition.	
08.11	A single named color MAY be implemented in an OpenXPS Document using an associated ICC Named Color type profile.	15.2.6
08.12	An the case of an ICC Named Color type profile the namedColor2Tag MAY be populated with specific device color values for each named color.	15.2.6
08.13	A consumer incapable of supporting a particular ICC profile tag that is optional in both ICC and OpenXPS MAY treat this tag as a user-defined custom tag, and therefore ignore it.	15.3.8
08.14	When no usable profile is present a consumer MAY choose to instantiate an error condition.	15.3.8
08.15	ICC profiles embedded in any image format (according to the restrictions of the image file format) with any color space.	15.1.8
08.16	OpenXPS producers MAY include ICC profiles for sRGB and scRGB color spaces.	15.1.8
08.17	An ICC profile MAY contain private tags.	15.1.8
08.18	Implementations MAY act on private tags.	15.1.8
08.19	Producers and consumers MAY support N-Channel colors in image data, using the TIFF image format.	15.1
08.20	A consumer MAY use the profile to obtain the encoded name of the named color.	15.2.6
08.21	A consumer MAY use the encoded name of a named color to lookup a device-specific color value for the named color.	15.2.6
08.22	A consumer MAY use the profile to obtain the encoded name of the named color.	15.2.6, 15.3.6
08.23	A consumer MAY use the encoded name of a named color to lookup a device-specific color value for the named color.	15.2.6, 15.3.6
08.24	A consumer MAY use the encoded name of a named color to lookup a device-specific color value for the named color	15.3.6
08.25	Consumers MAY use the ASCII name in the ICC profile or MAY compute a color approximation using a specified color value in the ICC profile; the results of these two methods MAY differ significantly.	15.3.6
08.26	A document registration named color identified in a PrintTicket MAY occur in an OpenXPS Document using the single named color and monochrome profile with tint LUT syntax.	15.4

1 **~~E.10~~F.10 Document Structure and Interactivity**2 **~~E.10.1~~F.10.1 MUST Conformance Requirements**3 *Table F-24. Document structure MUST conformance requirements*

ID	Rule	Reference
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
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
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
M9.4	Relative internal hyperlinks between FixedPage parts MUST specify, at a minimum, the named address relative to the FixedDocument part.	16.2.1
M9.5	In order to be addressable by either a hyperlink or the document outline, the named address MUST appear in the <PageContent.LinkTargets> element in the fixed document.	16.2.1
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
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
M9.8	Internal references MUST specify a page address relative to the fixed document sequence.	16.2.2
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
M9.10	The Name attribute MUST NOT be specified on any children of a <ResourceDictionary> element.	16.2.3
M9.11	The FragmentName attribute MUST be unique within the scope of the story.	16.1.1.6
M9.12	A <StoryBreak> element MUST NOT be included in a position other than the first or last child element of a <StoryFragment> element.	16.1.2
M9.13	A <TableRowGroupStructure> element is REQUIRED in order to specify a set of <TableRowStructure> elements.	16.1.2.7
M9.14	If specified, the Name value MUST meet the following requirements: The initial character MUST be an underscore character or a letter, that is, it falls within the Lu, Ll, Lo, Lt, and Nl categories. Trailing characters MUST be an underscore character or a letter or number, that is, they fall within the Lu, Ll,	16.2.3

---

Lo, Lt, NI, Mn, Mc, and Nd categories.

---

1 **E.10.2 F.10.2 SHOULD Conformance Requirements**

2 *Table F-25. Document structure SHOULD conformance requirements*

ID	Rule	Reference
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
S9.2	This requirement was removed prior to Edition 1 of this Standard.	
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
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
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
S9.6	If consumers enable user interactivity, they SHOULD support hyperlink activation and addressing.	16.2
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 OpenXPS Document, consumers SHOULD navigate to the URI.	16.2.1
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
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
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
S9.11	A hyperlink destination in the same fixed document SHOULD be expressed as a relative URI.	16.2.4
S9.12	This requirement was removed prior to Edition 1 of this Standard. If selection is supported, consumers SHOULD provide a visual cue over or around selected elements.	

S9.13	Selection order within an OpenXPS Document SHOULD follow reading order.	16.3
S9.14	In the absence of document structure provided in the OpenXPS Document, consumers SHOULD, at minimum, rely on the markup order to determine reading order.	16.4.1
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
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
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
S9.18	Producers SHOULD order <StoryFragment> elements in each StoryFragments part in their intended reading order.	16.4.1
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
S9.20	Producers SHOULD order <Story> elements in the DocumentStructure part in their intended reading order.	16.4.1
S9.21	Producers SHOULD order <StoryFragmentReference> elements within a <Story> element in their intended reading order.	16.4.1
S9.22	A screen reader consumer SHOULD read the document according to its reading order.	16.4.2
S9.23	A screen reader SHOULD use the UnicodeString attribute of each <Glyphs> element to determine the text to read.	16.4.2
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 OpenXPS Document.	16.4.2
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
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
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

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
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
S9.30	Children of <VisualBrush> elements SHOULD NOT be referenced by document structure markup.	16.1.2.13

1 **~~E.10.3~~F.10.3 OPTIONAL Conformance Requirements**

2 *Table F-26. Document structure OPTIONAL conformance requirements*

ID	Rule	Reference
O9.1	Producers MAY choose to add document structure information to OpenXPS Documents. Consumers MAY ignore any authored document structure or hyperlinks.	Clause 16
O9.2	Producers MAY provide either the document outline or the document content, or both; consumers MAY ignore either or both.	16.1
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
O9.4	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
O9.5	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
O9.6	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
O9.7	Consumers MAY ignore the Name attribute.	16.2.3
O9.8	Consumers MAY ignore the FixedPage.NavigateUri attribute.	16.2.4
O9.9	Viewing consumers that support interactivity MAY support selection and copying.	16.3
O9.10	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
O9.11	In the absence of document structure information provided in the OpenXPS Document, consumers MAY infer the reading order from the position of	16.4.1

	elements on the page.	
O9.12	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
O9.13	Screen readers MAY inspect the Indices attribute to resolve potential ambiguities in the UnicodeString attribute.	16.4.2
O9.14	The <DocumentStructure> element MAY contain a single <DocumentStructure.Outline> element and zero or more <Story> elements	16.1.1.1
O9.15	A <StoryFragment> element MAY be identified with a FragmentName attribute to distinguish it from other fragments for the same story on a single page.	16.1.2.2
O9.16	The <TableCellStructure> element defines the appearance of a table cell. It MAY contain nested <TableStructure> elements	16.1.2.9
O9.17	The FixedPage.NavigateUri attribute is OPTIONAL.	16.2.4

## 1 ~~E.11~~F.11 **OpenXPS Document Package Features**

### 2 ~~E.11.1~~F.11.1 **MUST Conformance Requirements**

3 *Table F-27. OpenXPS Document package feature MUST conformance requirements*

ID	Rule	Reference
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
M10.2	Consumers MUST be able to consume packages regardless of their interleaving structure.	17.1.3
M10.3	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Consumers that lack the resources to process a part MUST instantiate an error condition.	
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
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
M10.6	The DiscardControl part MUST NOT reference itself.	17.1.4.1
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

M10.8	All producers and consumers signing and verifying signatures for end users or applications MUST adhere to the OpenXPS Document signature policy, and producers and consumers MUST interpret digital signatures consistently.	17.2.1
M10.9	Consumers MUST NOT prevent an end user from taking an action solely because doing so will invalidate a signature.	17.2.1
M10.10	An OpenXPS Document MUST be considered signed according to the OpenXPS Document signing policy, regardless of the validity of that signature, if the signing rules described in §17.2.1.1 are observed. The following parts MUST be signed: All FixedDocument parts referenced in the markup of the FixedDocumentSequence part; FixedDocumentSequence part that is the target of the Start Part package relationship; All FixedPage parts referenced by all signed FixedDocument parts; <SignedInfo> portion of the Digital Signature XML Signature part containing this signature; All parts associated with each signed FixedPage part by means of a Required Resource relationship; All DocumentStructure parts associated via a Document Structure relationship with all signed FixedDocument parts; All StoryFragments parts associated via Story Fragments relationship with all signed FixedPage parts; All SignatureDefinitions parts associated via a Signature Definitions relationship with any signed FixedDocument part; All Thumbnail parts associated via a Thumbnail relationship from the package root or with any signed FixedPage or FixedDocument part	17.2.1.1
M10.11	An OpenXPS Document MUST NOT be considered signed according to the OpenXPS 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
M10.12	An OpenXPS Document digital signer MUST NOT sign an OpenXPS 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
M10.13	An OpenXPS Document digital signature MUST be <a href="#">treated</a> <del>shown</del> 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
M10.14	An OpenXPS Document digital signature MUST be shown as a broken digital signature if <a href="#">it is not an incompliant digital signature and it violates any of the signing rules described above regarding parts or relationships that MUST be signed, and</a> it is not an incompliant digital signature, but the signature fails the signature validation routines described in the OPC.	17.2.1.2
M10.15	An OpenXPS 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
M10.16	An OpenXPS 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
M10.17	To prohibit additional signatures in an OpenXPS Document, the signing	17.2.1.3



	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.	
M10.18	OpenXPS Document signatures MUST NOT refer to a remote certificate store. All certificates MUST be stored in the OpenXPS Document either as a Certificate part or in the Digital Signature XML Signature part.	17.2.1.4
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
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
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
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
M10.23	There MUST NOT be more than one DiscardControl package relationship.	17.1.4.1
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
M10.25	An OpenXPS Document digital signer MUST NOT sign a PrintTicket part if it does not fully understand the PrintTicket content.	17.2.1.1
M10.26	If the SignatureDefinitions part exists, it MUST contain only one <SignatureDefinitions> element.	17.2.2.1
M10.27	If the SignatureDefinitions part exists, there MUST be <i>at least</i> one <SignatureDefinition> element.	17.2.2.2
M10.28	The SpotID attribute is REQUIRED.	17.2.2.2.1
M10.29	The value of this attribute MUST be globally unique to ensure that a Signature part can be linked to only one <SignatureDefinition> element.	17.2.2.2.1

1 **E.11.2F.11.2 SHOULD Conformance Requirements**

2 *Table F-28. OpenXPS Document package feature SHOULD conformance requirements*

ID	Rule	Reference
S10.1	When interleaving, the Content Types stream SHOULD be interleaved according to the recommendations in the OPC Standard.	17.1
S10.2	When interleaving, PrintTicket parts SHOULD be written to the package before the part to which they are attached.	17.1
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

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 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
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	<a href="#">This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.</a> DiscardControl parts that are not well-formed SHOULD NOT be processed and an error condition SHOULD NOT be instantiated.	<del>17.1.4.1</del>
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
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
S10.11	Consumers SHOULD inform the end user if an action they are going to take will invalidate an existing signature.	17.2.1
S10.12	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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	
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
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 OpenXPS Document to be signed.	17.2.2.3
S10.15	It is RECOMMENDED that a signature spot be a clickable area used to launch the digital signing process.	17.2.2.3

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
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
S10.18	Head-first OpenXPS Document consumers SHOULD attempt to detect inconsistent packages as soon as possible and SHOULD instantiate an error condition, even if they have already processed the pages that resulted in the error.	17.1
S10.19	The viewing consumer SHOULD use the values specified in the <SpotLocation> element to place a signature spot.	17.2.2.3
S10.20	The relationships for the DiscardControl part and the StartPart SHOULD both be written in the first piece of the package relationship part, and that piece SHOULD be before the first FixedPage part in the package.	17.1
S10.21	The piece of the DiscardControl part that includes a Discard element with a SentinelPage attribute referencing a FixedPage part SHOULD be written to the package before that FixedPage part.	17.1
S10.22	Consumers that support printing of signed documents SHOULD support control through PrintTicket settings pertaining to the treatment of OpenXPS Documents with invalid or questionable signatures.	17.2.1.5
<a href="#">S10.23</a>	<a href="#">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 or until the end of the package is reached (indicating an error condition); if the end of the package is reached the consumer SHOULD instantiate an error condition.</a>	17.1.3

1 **~~E.11.3~~F.11.3 OPTIONAL Conformance Requirements**

2 *Table F-29. OpenXPS Document package feature OPTIONAL conformance requirements*

ID	Rule	Reference
O10.1	Interleaving is OPTIONAL.	17.1
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
O10.3	Consumers MAY discard FixedPage parts once they have been processed.	17.1.3
O10.4	Consumers MAY discard FixedDocument and FixedDocumentSequence parts after all their child elements and their closing tags have been processed.	17.1.3
O10.5	In the absence of explicit directives to the contrary, consumers MAY discard parts as directed by the DiscardControl part.	17.1.3
O10.6	Some producers (typically drivers) MAY choose a suitable interleaving order by modeling the resource management behavior of the consumer.	17.1.4

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
O10.8	An OpenXPS 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
O10.9	An OpenXPS 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
O10.10	OpenXPS Documents MAY be signed more than once.	17.2.1.3
O10.11	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. Producers MAY include the JobDigitalSignatureProcessing setting in the job-level PrintTicket within the OpenXPS Document content.	
O10.12	The SpotID attribute of the <SignatureDefinition> element MAY be used to link to an existing signature.	17.2.2.2.1
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
O10.14	The <SigningLocation> element MAY be set by the original producer of the OpenXPS Document or by the signing party at the time of <a href="#">requesting a signature signing</a> .	17.2.2.6
O10.15	The <SpotLocation> element is OPTIONAL.	17.2.2.3
O10.16	The following parts MAY be signed: CoreProperties, Digital Signature Certificate, Digital Signature Origin, DiscardControl, and PrintTicket.	17.2.1.1

## ~~E.12~~[F.12](#) Rendering Rules

### ~~E.12.1~~[F.12.1](#) MUST Conformance Requirements

Table F-30. Rendering rules MUST conformance requirements

ID	Rule	Reference
M11.1	Producers MUST generate OpenXPS 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 OpenXPS Documents	Clause 18
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
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
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

M11.5	If a consumer encounters markup with characteristics outside its implementation-defined limits, it MUST instantiate an error condition	18.2
M11.6	The alpha information in TIFF images using an ExtraSamples tag value of 1 and in <del>Windows Media Photo</del> JPEG XR images using pixel formats <del>WICPixelFormat32bppPBGRA</del> , <del>WICPixelFormat64bppPRGBA</del> or <del>WICPixelFormat128bppPRGBAFloat</del> MUST be interpreted as pre-multiplied alpha information.	18.4.1
M11.7	Composition MUST have the same effect as the application of the rules in §18.5, in sequence.	18.5
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
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
M11.10	The Width and Height values specified in the Viewbox and Viewport attributes of an <ImageBrush> or <VisualBrush> element MUST NOT be negative.	18.1.3

## 1 ~~E.12.2~~F.12.2 SHOULD Conformance Requirements

2 Table F-31. Rendering rules SHOULD conformance requirements

ID	Rule	Reference
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
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
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, 18.6.12
S11.4	When rendering geometries, consumers SHOULD render curves so they appear smooth from a normal viewing distance.	18.1.5
S11.5	When no anti-aliasing is used, abutting shapes that share	18.1.7

---

	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.	
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
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
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 OpenXPS Documents that stay within these implementation limits.	18.2
S11.9	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes.  Coordinates are real numbers and SHOULD be computed with at least single floating point precision.	
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
S11.11	Gradients SHOULD be rendered according to the guidelines described in §18.3.	18.3
S11.12	Consumers SHOULD pre-process gradient stops for all gradients using the steps described in §18.3.1.1.	18.3.1.1
S11.13	If any gradient stops use an sRGB or scRGB color specification consumers SHOULD blend colors between gradient stops in the color space indicated by the ColorInterpolationMode attribute of the gradient brush, unless a PrintTicket setting provides an alternative blending color space that the consumer understands.  If none of the gradient stop elements uses an sRGB or scRGB color specification and the consumer understands the blending color space PrintTicket setting, the blending color space PrintTicket setting SHOULD be used.	18.3.1.2
S11.14	If a ColorInterpolationMode value of SRgbLinearInterpolation	18.3.1.2

---

	is used, the BLEND() function SHOULD convert the color values to sRGB first, and then perform a linear interpolation between them.	
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
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
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
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	18.6.3

	exchanged).	
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
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, <a href="#">after application of relevant transforms</a> , the same StrokeThickness attribute value, and consumers aware of this convention SHOULD render such a stroke no thinner than the thinnest visible line that <a href="#">a bi-tonal consumer supports without dropouts</a> <a href="#">or an anti-aliasing consumer can represent as a solid line</a> . <a href="#">In the particular case of StrokeThickness attribute value of "0" the stroke SHOULD be rendered with a 1-pixel thickness if the nominal stroke width convention applies.</a>	18.6.12
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
S11.33	Consumers SHOULD render an element filled with a linear gradient brush such that the appearance is the same as if the steps described in §18.3.2 had been taken.	18.3.2
S11.34	Consumers SHOULD render an element filled with a radial gradient brush such that the appearance is the same as if the steps described in §18.3.3 had been taken.	18.3.3



1 **E-12.3 F.12.3 OPTIONAL Conformance Requirements**2 *Table F-32. Rendering rules OPTIONAL conformance requirements*

ID	Rule	Reference
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
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
O11.3	Devices MAY use sub-pixel masking.	18.1.4
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
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, 18.5
O11.6	Consumers MAY apply pixel placement rules optimized for character rendering to individual glyphs in a <Glyphs> element.	18.1.6
O11.7	Behavior of blending with very close gradient stops MAY vary in an implementation-defined manner (see S11.16).	18.3.1.2
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
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
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 by ignoring the superluminous portion of each color channel value.	18.4.1
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	18.5.1

	contained area, the canvas and all of its child and descendant elements MAY be skipped.	
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

## ~~E.13~~F.13 Additional Conformance Requirements

### ~~E.13.1~~F.13.1 MUST Conformance Requirements

Table F-33. Additional MUST conformance requirements

ID	Rule	Reference
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.	-
M12.2	A FixedDocument part MUST NOT be referenced more than once by a FixedDocumentSequence part.	-
M12.3	A FixedPage part MUST NOT be referenced more than once <i>in total</i> , throughout all	-

	FixedDocument parts.	
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.	-
M12.5	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 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.	
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.	-
M12.7	<del>The content types in the tables below MUST NOT include parameters.</del> If a consumer encounters the presence of parameters on these content types <a href="#">in the tables in this subclause</a> when the affected part is accessed it MUST instantiate an error condition.	D.2
<a href="#">M12.8</a>	<a href="#">The content types in the tables in this subclause MUST be used by producers without parameters.</a>	D.2

## 1 ~~E.14~~F.14 3D Graphic Content

### 2 ~~E.14.1~~F.14.1 MUST Conformance Requirements

#### 3 Table F-34. 3D Graphic Content MUST conformance requirements

ID	Rule	Reference
M13.1	A consumer that renders OpenXPS pages for printing on a 2D medium MUST NOT read and render the 3D content to a 2D representation unless at least one of the following is true: It has been explicitly configured to do so by the user; or it provides explicit feedback to the user that the 3D content is being used rather than the 2D representation.	GA
M13.2	The X3D file MUST conform to one of the following standards meeting the X3D "Interoperability" conformance level: <a href="#">ISO/IEC FCD 19775-1r1:200x</a> , <a href="#">ISO/IEC 19776-1:2005</a> , <a href="#">ISO/IEC 19776-3:2007</a> .	GA
M13.3	3D Producers MUST use a <Brush3D> element within a <Path.Fill> element, with Markup Compatibility, to place the 3D content on a page within a defined Viewport.	GA

M13.4	3D producers MUST define a conforming <AlternateContent.Fallback> element within the parent <Path> element of the <Brush3D> element. The <AlternateContent.Fallback> element MUST contain a 2D representation for viewing and printing of the 3D content.	GA
M13.5	3D Producers MUST define the AlternateContent.Fallback that is visually representative of the default 3D viewpoint.	GA
M13.6	Implementations that modify the 3D default viewpoint MUST update the AlternateContent.Fallback to match that 3D viewpoint prior to printing or saving the file	GA
M13.7	3D Consumers MUST display at least the AlternateContent.FallBack.	GA
M13.8	The active 3D window display MUST be contained within the defined Brush3D.Viewport.	GA

1 **E.14.2F.14.2 SHOULD Conformance Requirements**

2 *Table F-35. 3D Graphic Content SHOULD conformance requirements*

ID	Rule	Reference
S13.1	3D Consumers SHOULD allow user interaction to navigate and play animations present in the 3D graphics content.	GA
S13.2	3D Producers generating X3D model content SHOULD follow these rules: 1. Use Triangle based X3D elements to facet the model. 2. Use positive values for all X Y Z coordinates. 3. Define a face normal for all Triangle faces. 4. Define Triangle faces counter clockwise (right-hand rule) when facing outwards or upwards in ground terrain models. 5. Ensure that all triangle faces share two points with their neighboring faces.	GA
S13.3	This requirement was removed prior to Edition 1 of this Standard; its description is retained here for historical purposes. 3D Producers SHOULD define equivalent AlternateContent.Fallback sufficient for 2D viewing and printing	G
S13.4	3D Consumers SHOULD display the X3D 3D content rendered to display the default 3D viewpoint and perspective as defined in the X3D part.	GA
<a href="#">S13.5</a>	<a href="#">3D consumers SHOULD display an active, navigatable 3D window. If UI controls are located outside the viewport, no resizing of viewport SHOULD cause repagination or alteration of the page fixed format</a>	G

1 **~~E.14.3~~F.14.3 OPTIONAL Conformance Requirements**

2 *Table F-36. 3D Graphic Content OPTIONAL conformance requirements*

ID	Rule	Reference
O13.1	An OpenXPS producer MAY include three-dimensional (3D) graphics within an OpenXPS package.	GA

3 **F.15 Recommended File Name Extension and Content Types**

4 **F.15.1 MUST Conformance Requirements**

5 *Table F-37. Recommended File Name Extension and Content Types MUST conformance requirements*

<u>ID</u>	<u>Rule</u>	<u>Reference</u>
<u>M14.1</u>	<u>To avoid conflicts between OpenXPS Documents defined in this Standard and legacy formats, producers MUST NOT create OpenXPS Documents with filenames that end in the uppercase, lowercase, or mixed-case sequence .xps.</u>	E.1

6 **F.15.2 SHOULD Conformance Requirements**

7 *Table F-38. Recommended File Name Extension and Content Types SHOULD conformance requirements*

<u>ID</u>	<u>Rule</u>	<u>Reference</u>
<u>S14.1</u>	<u>Implementations are anticipated for multiple operating systems, including operating systems that use the concept of filename extension and/or content type to identify the format of files for processing. When required by such systems, and to enable interoperability with such systems, implementations SHOULD use a filename extension or termination sequence of .oxps and a content type of application/oxps.</u>	E.1
<u>S14.2</u>	<u>Producers SHOULD include an XML comment immediately following the start-tag of the FixedPage element. This comment SHOULD include details of the organization, product, and version that created the content with the intention that this could be used as an aid in the diagnosis of problem content.</u>	E.2

8 **End of informative text.**



## 1 **F.G. 3D Graphic Content**

2 An OpenXPS producer MAY include three-dimensional (3D) graphics within an OpenXPS  
3 package [O13.1]. The 3D graphics content provides OpenXPS documents with animatable  
4 3D models to supplement the document text providing a visually richer user experience in  
5 3D Consumers for purposes such as, but not limited to:

- 6 1. 3D model examination and walkthroughs.
- 7 2. Animations depicting assembly instructions.
- 8 3. Animations showing usage instructions.
- 9 4. Animations depicting proposed phased building construction projects.

10 3D content is included in such a way that an alternative representation that is suitable for use  
11 in two-dimensional (2D) rendering is provided for a consumer that does not support  
12 3D content; e.g., for printing on paper.

13 This Annex does not introduce any additional requirements for a producer or consumer that  
14 does not support 3D content.

15 A consumer that renders OpenXPS pages for printing on a 2D medium MUST NOT read and  
16 render the 3D content to a 2D representation unless at least one of the following is true  
17 [M13.1]:

- 18 1. It has been explicitly configured to do so by the user; or
- 19 2. It provides explicit feedback to the user that the 3D content is being used rather than  
20 the 2D representation.

21 An instance of 3D graphics content is created by placing a conformant X3D file within the  
22 OpenXPS document OPC Package. The X3D file MUST conform to one of the following standards  
23 meeting the X3D "Interoperability" conformance level [M13.2]: ISO/IEC FCD 19775-1r1:200x,  
24 ISO/IEC 19776-1:2005, or ISO/IEC 19776-3:2007.

25 Markup compatibility is used to encapsulate the 3D content and its 2D alternative  
26 representation.

27 A 3D Content Capable Producer (3D Producer) is defined as an OpenXPS producer that  
28 understands the "<http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0>" namespace.

29 A 3D Content Capable Consumer (3D Consumer) is defined as an OpenXPS consumer that  
30 understands the "<http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0>" namespace.

31 3D Producers generating X3D model content SHOULD follow these rules [S13.2]:

- 32 1. Use Triangle based X3D elements to facet the model.
- 33 2. Use positive values for all X Y Z coordinates.
- 34 3. Define a face normal for all Triangle faces.
- 35 4. Define Triangle faces counter clockwise (right-hand rule) when facing outwards or  
36 upwards in ground terrain models.

5. Ensure that all triangle faces share two points with their neighboring faces.

3D Producers MUST use a <Brush3D> element within a <Path.Fill> element, with Markup Compatibility, to place the 3D content on a page within a defined Viewport [M13.3].

3D producers MUST define a conforming <AlternateContent.Fallback> element within the parent <Path> element of the <Brush3D> element. The <AlternateContent.Fallback> element MUST contain a 2D representation for viewing and printing of the 3D content [M13.4].

3D Producers MUST define the AlternateContent.Fallback that is visually representative of the default 3D viewpoint [M13.5].

Implementations that modify the 3D default viewpoint MUST update AlternateContent.Fallback to match that 3D viewpoint prior to printing or saving the file [M13.6].

*Example G-1. 3D graphics content in FixedPage.fpage*

```

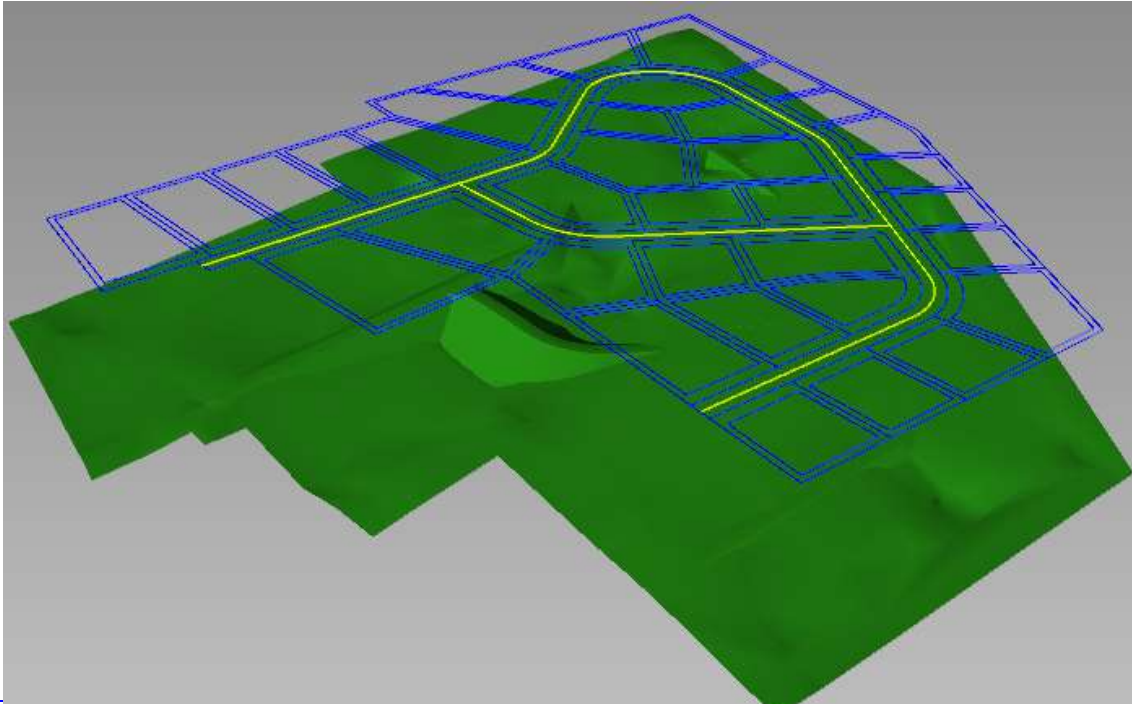
<FixedPage
  xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
  Height="1056" Width="816" xml:lang="und"
  xmlns:x3dfp="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0"
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006">
  <Canvas Name="dwfresource_1" RenderTransform="96, -0, -0,96,0,0">
    <Path Data="M0,0 L8.5,0 L8.5,11 L0,11 z">
      <Path.Fill>
        <mc:AlternateContent>
          <mc:Choice Requires="fp3d">
            <fp3d:Brush3D fp3d:Source3D="olympus.x3d"
              fp3d:Viewport="0,2.3125,
                8.5,8.6875" fp3d:ViewportUnits="Absolute" fp3d:Viewbox="0,0,
                640,640" fp3d:ViewboxUnits="Absolute"/>
          </mc:Choice>
          <mc:Fallback>
            <ImageBrush ImageSource="ProxyGraphics.png" Viewport="0,2.3125,
              8.5,8.6875" ViewportUnits="Absolute" Viewbox="0,0, 640,640"
              ViewboxUnits="Absolute"/>
          </mc:Fallback>
        </mc:AlternateContent>
      </Path.Fill>
    </Path>
  </Canvas>
</FixedPage>

```

3D consumers SHOULD display an active, navigatable 3D window. If UI controls are located outside the viewport, no resizing of viewport SHOULD cause repagination or alteration of the page fixed format [S13.5].

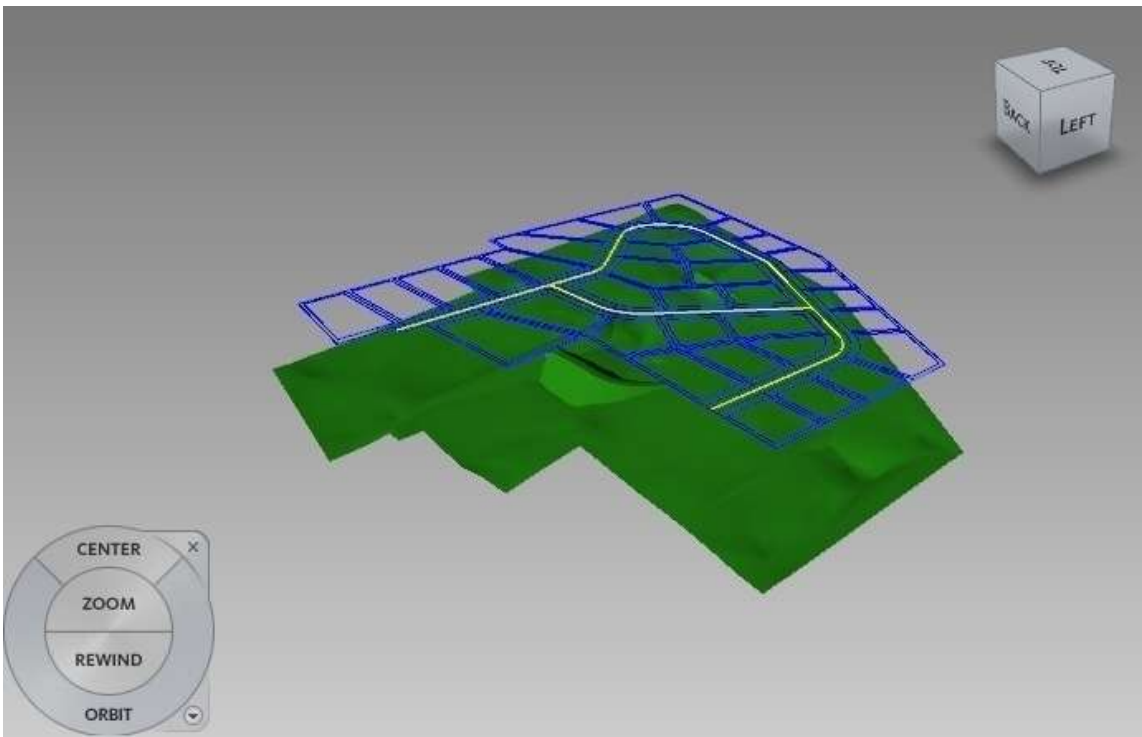
Non-3D Consumers will display AlternateContent.Fallback; ~~that is, in this example, the following 2D image~~ for example:





1

2 For this example, 3D Consumers should display an active, navigatable 3D window similar to the  
3 image below.



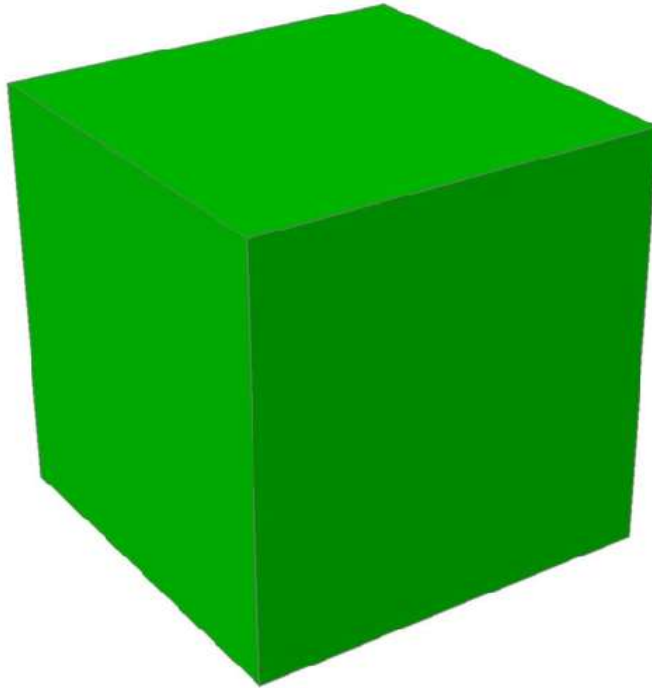
4

5 *end example]*

```

1  Example G-2. 3D graphics content in FixedPage.fpage
2  <FixedPage
3  |  xmlns="http://schemas.microsoft.com/xps/2005/06schemas.openxps.org/oxps/v1.0"
4  |  Height="1056"
5  |  Width="816" xml:lang="und"
6  |  xmlns:x3dfp="http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0"
7  |  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006">
8  |  <Canvas Name="dwfresource_1" RenderTransform="96, -0, -0,96,0,0">
9  |  |  <Path>
10 |  |  |  <mc:AlternateContent>
11 |  |  |  |  <mc:Choice Requires="x3d">
12 |  |  |  |  |  <Path.Fill>
13 |  |  |  |  |  |  <fp:x3d:Brush3D fp:x3d:Source3D="olympus3dcube.x3d"
14 |  |  |  |  |  |  fp:x3d:Viewport="0,2.3125,
15 |  |  |  |  |  |  8.5,8.6875" fp:x3d:ViewportUnits="Absolute" fp:x3d:Viewbox="0,0,
16 |  |  |  |  |  |  640,640" fp:x3d:ViewboxUnits="Absolute"/>
17 |  |  |  |  |  </Path.Fill>
18 |  |  |  |  </mc:Choice>
19 |  |  |  |  <mc:Fallback>
20 |  |  |  |  |  <Path.Data>
21 |  |  |  |  |  |  <PathGeometry>
22 |  |  |  |  |  |  |  <PathFigure StartPoint="120,160" IsClosed="true">
23 |  |  |  |  |  |  |  >
24 |  |  |  |  |  |  |  |  <PolyBezierSegment Points="70,0 120,160 170,80 120,0 70,160
25 |  |  |  |  |  |  |  |  120,160"/>
26 |  |  |  |  |  |  |  </PathFigure>
27 |  |  |  |  |  |  |  <PathFigure StartPoint="120,160" IsClosed="true">
28 |  |  |  |  |  |  |  |  <PolySegment Points="170,0 220,160 270,80 120,160"/>
29 |  |  |  |  |  |  |  </PathFigure>
30 |  |  |  |  |  |  |  <PathFigure StartPoint="120,160" IsClosed="true">
31 |  |  |  |  |  |  |  |  <PolySegment Points="30,0 60,80 10,40
32 |  |  |  |  |  |  |  |  120,0"/>
33 |  |  |  |  |  |  |  </PathFigure>
34 |  |  |  |  |  </PathGeometry>
35 |  |  |  |  </Path.Data>
36 |  |  |  |  </mc:Fallback>
37 |  |  |  </mc:AlternateContent>
38 |  |  </Path>
39 |  </Canvas>
40 | </FixedPage>
41 | end example]

```



- 1 .
- 2 3D Consumers MUST display at least the AlternateContent.FallBack [M13.7].
- 3 3D Consumers SHOULD display the X3D 3D content rendered to display the default
- 4 3D viewpoint and perspective as defined in the X3D part [S13.4].
- 5 3D Consumers SHOULD allow user interaction to navigate and play animations present in the
- 6 3D graphics content [S13.1]. The active 3D window display MUST be contained within the
- 7 defined Brush3D.Viewport [M13.8].

targetNamespace: <http://schemas.openxps.org/openxps/3d/2008oxps-3d/v1.0>

## 8 **F.1G.1** Brush3D

9 Table 13–1. Brush types is extended by the addition of the following brush type:

10

Name	Description
3D Content Brush	Fills a region with a 3D graphics model (Annex A)

11 In the following diagrams and text the prefix “fp” refers to the FixedPage name space. See §19

12 for definitions of those elements and attributes. xx

1

<p>diagram</p>													
<p>namespace</p>	<p><a href="http://schemas.openxps.org/openxps/3d/2008/xps-3d/v1.0">http://schemas.openxps.org/openxps/3d/2008/xps-3d/v1.0</a></p>												
<p>type</p>	<p>extension of CT_Brush3D</p>												
<p>properties</p>	<p>content complex</p>												
<p>children</p>	<p><a href="#">xps4:ImageBrush.Transform</a></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><a href="#">Source3D</a></td> <td>ST_UriImage3D</td> <td>required</td> <td></td> <td></td> <td><a href="#">-Specifies the URI of an X3D</a></td> </tr> </tbody> </table>	Name	Type	Use	Default	Fixed	annotation	<a href="#">Source3D</a>	ST_UriImage3D	required			<a href="#">-Specifies the URI of an X3D</a>
Name	Type	Use	Default	Fixed	annotation								
<a href="#">Source3D</a>	ST_UriImage3D	required			<a href="#">-Specifies the URI of an X3D</a>								

	<p><a href="#">x:Key</a></p>			<p><a href="#">resource</a>. The <a href="#">URI MUST refer to parts in the package [M2.1]</a>.</p> <p>-Specifies a name for a resource in a resource dictionary. <a href="#">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]</a>.</p>
	<p><a href="#">Transform</a></p>	<p><a href="#">oxps4:ST_RscRefMatrix</a></p>		<p>-Describes the matrix transformation applied to the coordinate space of the brush. The <a href="#">Transform property is concatenated with the current effective render transform to yield an effective render transform local to the brush</a>. The <a href="#">viewport for the brush is transformed using that local effective render transform</a>.</p>
	<p><a href="#">Viewbox</a></p>	<p><a href="#">oxps4:ST_ViewBox</a></p>	<p>required</p>	<p>-Specifies the position and dimensions of the brush's source content. <a href="#">Specifies four comma-separated real numbers (x, y, Width, Height), where width and height are non-negative</a>.</p>

	<p><a href="#">Viewport</a></p>	<p>oxps4:ST_ViewBox</p>	<p>required</p>		<p>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.</p> <p>-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.</p>
	<p><a href="#">ViewboxUnits</a></p>	<p>oxps4:ST_ViewUnits</p>	<p>required</p>	<p>Absolute</p>	<p>-Specifies the relationship of the viewbox coordinates to the containing coordinate space</p>
	<p><a href="#">ViewportUnits</a></p>	<p>oxps4:ST_ViewUnits</p>	<p>required</p>	<p>Absolute</p>	<p>-Specifies the relationship of the viewport coordinates to the containing coordinate space.</p>
<p>source</p>	<p>&lt;xs:element name="Brush3D"&gt;</p>				

	<pre>&lt;xs:complexType&gt;   &lt;xs:complexContent&gt;     &lt;xs:extension base="CT_Brush3D"/&gt;   &lt;/xs:complexContent&gt; &lt;/xs:complexType&gt; &lt;/xs:element&gt;</pre>
--	--

- 1 *[Example:*  
2     <Brush3D Source3D="olympus.x3d" Viewport="0,2.3125, 8.5,8.6875"  
3     ViewportUnits="Absolute" Viewbox="0,0, 640,640" ViewboxUnits="Absolute"/>  
4 This example shows the required attributes and example values. The Source3D attribute is a  
5 URI that must resolve to a conforming X3D file contained within the OpenXPS OPC Package.  
6 See §19 for definitions of Viewport, ViewportUnits, Viewbox, ViewBoxUnits and  
7 ImageBrush.Transform. *end example]*





## 1 **G.H. 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 ~~Windows Media Photo Microsoft Corporation. <http://www.microsoft.com/xps>~~

12 X3D Specification Web3D, Consortium. <http://www.web3d.org>



1 **H.I. 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

7 

---

8 **3**

9 3D Graphic Content .....453

10 

---

10 **A**

11 abbreviated geometry syntax ..... See geometry, abbreviated  
 12 syntax

13 accessibility .....47

14 document structure, enabled by .....211

15 image text alternative, long

16 described .....235

17 on canvas .....56

18 on path .....63

19 image text alternative, short

20 described .....235

21 on canvas .....56

22 on path .....63

23 importance of .....234

24 mentioned .....211

25 of text .....104

26 page elements, requirement to expose all .....234

27 reading order

28 document structure, dependent on .....234

29 fragment type, dependent on .....234

30 markup order, dependent on .....234

31 of page-centric application .....234

32 of story-centric application .....234

33 screen reader

34 considerations .....235

35 mentioned .....211, 234

36 alpha ..... See opacity

37 anti-aliasing

38 disabling of .....56

39 rendering of .....257

40 arc ..... See geometry, segment, arc

41 

---

41 **B**

42 bleed area ..... See page, bleed area

43 bookmark ..... See document structure, outline

44 brush .....117

45 alpha ..... See opacity

46 described .....117

47 gradient computations ..... 259–66

48 gradient stop

49 color, specifying ..... 156

50 described ..... 156

51 offset ..... 156

52 specifying ..... 156

53 image brush

54 described ..... **119**, 121

55 example ..... 121

56 image ..... See image

57 mentioned ..... 28, 61

58 source, specifying ..... 120

59 tile size and placement ..... See brush, view port

60 tile source ..... See brush, view box

61 linear gradient brush

62 color interpolation mode ..... 143

63 described ..... 144

64 end point ..... 143

65 example ..... 144

66 gradient stops, specifying ..... 148

67 mappng mode ..... 143

68 rendering ..... 261

69 specifying ..... 143

70 spread method

71 described ..... 145

72 Pad ..... 145

73 Reflect ..... 146

74 Repeat ..... 147

75 specifying ..... 143

76 start point ..... 143

77 opacity ..... See opacity

78 radial gradient brush

79 described ..... 151

80 example ..... 151

81 gradient center ..... 149

82 gradient origin ..... 149

83 gradient stops, specifying ..... 155

84 gradient x-radius ..... 149

85 gradient y-radius ..... 149

86 mappng mode ..... 149

87 rendering ..... 263

88 specifying ..... 149

89 spread method

1	described.....	152	55	opacity mask.....	See opacity mask
2	Pad.....	152	56	transformation.....	See transformation
3	Reflect.....	153	57	caret stop.....	See selection, caret stop
4	Repeat.....	154	58	CFF font.....	See font:CFF
5	specifying.....	149	59	circle.....	See geometry, segment, arc
6	solid color brush.....	118, 119	60	clipping	
7	tile		61	described.....	170
8	behavior.....	See brush, tile, mode	62	elements applied to.....	161, 162
9	image scaling.....	285	63	geometry, reference to.....	70
10	mode		64	of canvas.....	56, 170
11	described.....	132–42	65	of glyphs.....	92, 173
12	FlipX.....	137–38	66	of path.....	63, 172
13	FlipXY.....	141–42	67	rules.....	257
14	FlipY.....	139–40	68	CMYK.....	See color, color space, CMYK
15	mentioned.....	127	69	color	
16	None.....	132–34	70	alpha.....	See opacity
17	specifying.....	120, 123, <b>127</b>	71	blending.....	<b>9</b>
18	Tile.....	135–36	72	behavior described.....	209
19	placement.....	See brush, view port	73	blend color space for linear gradient brush.....	143
20	size.....	See brush, view port	74	for gradients.....	260
21	small tile rendering.....	285	75	implementation dependent.....	209
22	source.....	See brush, view box	76	brush, specifying for.....	See brush
23	transparent brush tiling.....	286	77	color profile	
24	transformation.....	See transformation	78	embedded in image . 28, 29, 30, 31, 32, 33, 196, 208, 427	
25	view box		79	ICC profile	
26	calculating source coordinates for images.....	127	80	color channels supported.....	197
27	described.....	127	81	described.....	196
28	example.....	128–31	82	grayscale image, usage with.....	197
29	larger than image.....	127, 131	83	parts.....	See parts
30	mapping to view port.....	120, 123, 458	84	profile types allowed.....	197
31	specifying for image brush.....	120	85	Windows Color System (WCS) profile	
32	specifying for visual brush.....	123	86	embedded in ICC profile.....	197
33	syntax.....	120, 123, 458	87	color separation.....	209
34	unit type.....	120, 123, 127	88	color space	
35	units, for images.....	120, <b>127</b> , 458	89	CMYK.....	196
36	view port		90	in images.....	195
37	described.....	127	91	in vector graphics.....	195
38	example.....	128–31	92	gray colors.....	196
39	placement precision.....	286	93	ICC profile	
40	specifying for image brush.....	120	94	version supported.....	197, 210, 425, 430, 432
41	specifying for visual brush.....	123	95	named color.....	<b>10</b> , 196
42	syntax.....	120, 123, 458	96	n-channel.....	196
43	unit type.....	120, 123, 127	97	in images.....	195
44	visual brush		98	in vector graphics.....	195
45	described.....	<b>123</b> , 124	99	scRGB.....	196
46	example.....	125	100	in images.....	195
47	visual.....	123, 124	101	in vector graphics.....	195
			102	spot colors.....	See color, color space, named colors
48	<b>C</b>		103	sRGB.....	196
49	canvas		104	in images.....	195
50	anti-aliasing control.....	56	105	in vector graphics.....	195
51	clipping.....	See clipping	106	support required.....	195
52	composing properties.....	58	107	support summarized.....	195
53	described.....	56	108	fidelity, improved.....	195
54	opacity.....	See opacity	109	raster image support	
			110	associating a color profile part.....	207
			111	CMYK.....	205

1	device color.....	207	55	relationships .....	<i>See relationships</i>
2	gray colors .....	204	56	request.....	<i>See digital signature, signature definitions</i>
3	named colors .....	206	57	signature definitions .....	<b>12</b>
4	n-channel .....	205	58	described .....	40, 249
5	scRGB.....	204	59	markup example.....	249
6	sRGB.....	203	60	mentioned.....	24
7	syntax		61	namespace .....	<i>See namespace, See namespace</i>
8	CMYK .....	200	62	relationship to .....	26
9	named color .....	202	63	sign by date and time .....	253
10	n-channel .....	201	64	signer name.....	250
11	scRGB.....	199	65	signing intent.....	252
12	sRGB.....	199	66	signing location .....	253
13	summarized .....	198	67	specifying .....	250
14	where used .....	197	68	spot ID .....	250, 251
15	color profile .....	<i>See color, color profile</i>	69	spot location.....	251
16	composability of properties..	<i>See OpenXPS Document format,</i>	70	signature policy	
17	properties, composability		71	conditions where policy does not apply.....	247
18	compression		72	described .....	246
19	image.....	<i>See image</i>	73	markup compatibility impact .....	248
20	package .....	<i>See Open Packaging Conventions, Standard</i>	74	parts to sign	
21	conformance		75	optional.....	247
22	inherited from Open Packaging Conventions.....	19, 400	76	required .....	246
23	of implementation .....	3	77	relationships to sign	
24	requirements tables .....	396–449	78	as a group, required.....	247
25	consumer .....	9	79	conditionally required.....	247
26	implementation burden .....	396–449	80	required .....	247
27	content area .....	<i>See page, content area</i>	81	signing rules .....	<b>12</b> , 246–48
28	content type .....	9	82	signing validity.....	248
29	namespace .....	<i>See namespace</i>	83	single signature .....	247
30	summarized.....	394	84	signature spot ....	<i>See digital signature, signature definitions</i>
31	usage of.....	23	85	signature status	
32	contour intersection point.....	9	86	broken .....	9
33	copy and paste.....	<i>See also selection</i>	87	broken digital signature .....	248
34	document structure, improved by .....	211	88	compliant .....	9
35	core properties .....	<i>See Open Packaging Conventions</i>	89	incompliant .....	9
36	curve .....	<i>See geometry, segment</i>	90	incompliant digital signature.....	248
			91	questionable.....	9
			92	questionable digital signature .....	248
			93	valid .....	9
			94	valid digital signature .....	248
37	<b>D</b>		95	discard control	
38	device.....	9	96	consumer considerations.....	243
39	device color.....	<i>See color, color space, device color</i>	97	elements .....	244
40	digital signature		98	markup example .....	244
41	certificate		99	namespace.....	<i>See namespace</i>
42	relationship to.....	26	100	part .....	<i>See parts</i>
43	store.....	249	101	reference, invalid.....	245
44	validity .....	248	102	reference, not yet encountered.....	245
45	co-signature .....	<i>See digital signature, signature definitions</i>	103	resource constraints, addressing .....	243
46	multiple signatures.....	248	104	sentinel page.....	245
47	namespace .....	<i>See namespace</i>	105	target resource .....	245
48	Open Packaging Conventions, extended from .....	246	106	usage of .....	244, 258
49	origin		107	document	
50	part .....	<i>See parts</i>	108	markup.....	<i>See OpenXPS elements, document-level</i>
51	relationship to.....	26	109	namespace.....	<i>See namespace</i>
52	parts .....	<i>See parts</i>	110	order of.....	49
53	printing.....	249	111	reference to .....	49
54	relationship to .....	26			

1	document conventions	13	58	markup elements summarized	218
2	diagram notes	13	59	markup example	224, 225
3	document outline	<i>See document structure, outline</i>	60	merging fragments	218–22
4	document roll-up	<i>See document, order of</i>	61	referencing every page element	224
5	document sequence, namespace	<i>See namespace</i>	62	relation to fixed page	211
6	document structure		63	story belonged to	223
7	constructed algorithmically	40	64	story, not belonging to any	224
8	constructed explicitly	40	65	story fragments	<i>See namespace, See namespace</i>
9	content	211	66	table	
10	content structure	9	67	cell	
11	document content	9	68	column span	229
12	figure	230	69	merging	218
13	list		70	row span	229
14	item		71	specifying	229
15	marker	230	72	row group, specifying	228
16	specifying	230	73	row, specifying	228
17	specifying	229	74	specifying	228
18	markup example	212	75	thread	<i>See document structure, story</i>
19	named element	10	76	usage of	40
20	canvas descendant elements	212	77	usage optional	211
21	described	211	78	driver	9
22	link target, optimizing location with	212	79	described	243
23	markup compatibility, updating for	231			
24	referencing each once	212			
25	specifying	231			
26	visual brush descendants prohibited	212			
27	namespace	<i>See namespace</i>			
28	naming page elements	161, 211	80	<b>E</b>	
29	outline	9, 211	81	effective coordinate space	<i>See layout, coordinate space</i>
30	language	<i>See language, of outline, See language, of</i>	82	elements	<i>See OpenXPS elements</i>
31	outline		83	error condition	4
32	levels	214	84	EXIF	
33	markup described	213	85	usage in JPEG	<i>See image, JPEG, EXIF</i>
34	markup example	214	86	usage in TIFF	<i>See image, TIFF, EXIF</i>
35	mentioned	40	87	extensibility of OpenXPS Documents	<i>See markup</i>
36	outline entry	213	88	compatibility	
37	target URI	214			
38	paragraph		89	<b>F</b>	
39	specifying	227	90	figure	
40	parts	<i>See parts</i>	91	illustration	<i>See document structure, figure</i>
41	relationships	<i>See relationships</i>	92	shape	<i>See geometry, figure</i>
42	section		93	fill	
43	specifying	227	94	algorithm	<i>See geometry, fill algorithm</i>
44	story	12	95	of path	<i>See path, fill brush</i>
45	described	40, 211	96	of stroke	<i>See path, stroke brush</i>
46	markup	215	97	find	47, 104, 111
47	markup example	217	98	fixed document sequence	<i>See document, order of</i>
48	story fragment, correlating to page	216	99	fixed page	<i>See page</i>
49	story fragment, reference to	216	100	fixed payload	<i>See OpenXPS Document format</i>
50	story fragment	12	101	FixedDocument part	10
51	break indicator	218, 227	102	FixedDocumentSequence part	10
52	content structure, contains	41, 211	103	FixedPage part	10
53	described	41, 211	104	font	
54	fragment name	223	105	CFF	35, 106
55	fragment name, uniqueness	216	106	compatibility encoding	38
56	fragment type	223	107	device font	92, 111
57	markup described	222, 223	108	embedding	35–38, 37
			109	extraction	36, 37

1	language impact on copy and paste	47
2	licensing rights	35, 37, 38
3	obfuscation	36, 37
4	algorithm	37
5	OpenType	35
6	parts	See parts
7	rasterization	36
8	relationships	See relationships, required resource
9	restricted editing	See relationships, restricted font
10	restricted font	27
11	sharing	35
12	subsetting	35, 36
13	TrueType	35
14	TrueType collection (TTC)	35, 412
15	Unicode encoding	35
16	usage of	35
<hr/>		
17	<b>G</b>	
18	geometry	
19	abbreviated syntax	73, 83–89, 417
20	algorithm	387
21	circle	See geometry, segment, arc
22	curve	See geometry, segment
23	described	69
24	figure	
25	closed	74, 82, 86
26	described	69
27	fill control	74
28	markup	74
29	reference to	71, 73
30	segments, composed of	69
31	start point	74, 84
32	stroking of segments	69
33	figures, composed of	69
34	fill algorithm	
35	described	72
36	EvenOdd	72
37	mentioned	69, 418
38	NonZero	73
39	specifying	71, 84
40	filled area	71
41	segment	
42	arc	75–78, 85
43	Bézier curve	79, 85
44	Bézier curve, quadratic	81, 85
45	Bézier curve, smooth	85, 87
46	line	80, 84
47	segment, degenerate	283
48	transformation	See transformation
49	usage described	70
50	glyphs	See text
51	gradient	See brush, See brush
52	graphics	See path
53	grouping markup	See canvas

54	<b>H</b>	
55	hairline	See stroke, hairline
56	hyperlink	
57	activation	231
58	addressability	
59	appearance in link targets	232
60	mentioned	51, 232
61	missing name, handling of	232
62	name	232
63	name uniqueness	232
64	of canvas	56, 232
65	of glyphs	92, 232
66	of page	53, 232
67	of path	63, 232
68	of visual brush contents	232
69	page number	232
70	document-level listing	See hyperlink, target
71	example	232
72	mentioned	211
73	overlapping behavior	231
74	source	
75	base URI	233
76	described	233
77	from canvas	56, 231
78	from glyphs	92, 231
79	from path	63, 231
80	inheritance of	231
81	specifying	161
82	support recommended	233
83	target	
84	addressability	See hyperlink, addressability
85	document-level definition of	52
86	external	231
87	internal	231
88	link target, specifying as	51
89	missing behavior	232
90	name	See name
91	relative target handling	232
92	relative target recommended	233
93	relative to document, at minimum	231
<hr/>		
94	<b>I</b>	
95	ICC	See color, color profile
96	image	
97	brush	See brush, image brush
98	JPEG	24, 28
99	APP markers	28, 29
100	CMYK	29
101	EXIF	28
102	naming	28
103	specification	28
104	parts	See parts
105	PNG	24, 28, 29
106	chunks	30

1	naming	29
2	specification	29
3	relationships	<i>See relationships, required resource</i>
4	resolution	127
5	resource	<i>See resource</i>
6	sharing	28
7	thumbnail	24
8	TIFF	24, 28, 30
9	alpha, associated	32
10	CCITT bilevel encoding	32
11	CMYK	32
12	compression	33
13	EXIF	33
14	features, supported	32
15	image file directory (IFD)	32
16	naming	30
17	specification	30, 33
18	tags, supported	30
19	tags, unsupported	32
20	tags, unsupported	33
21	variations, handling	33
22	types supported	28, 121
23	usage described	28, 61
24	JPEG XR	24, 28, 33
25	CMYK	34
26	features supported	33
27	grayscale	33
28	named color	34
29	naming	33
30	N-channel	34
31	profiled RGB	34
32	scRGB	34
33	specification	33
34	sRGB	34
35	implementation limits	258–59
36	implementation-defined behavior	10
37	ink area	<i>See page, content area</i>
38	interleaving	<i>See Open Packaging Conventions, interleaving</i>

---

**J**

40	JPEG	<i>See image, JPEG</i>
----	------	------------------------

---

**L**

42	language	
43	markup	48, 162
44	of canvas	56
45	of glyphs	92, 111
46	of outline	48, 213
47	of outline entry	48, 214
48	of page	53
49	of path	63
50	of resource	166
51	of signature definition	250
52	usage	47

53	layout	<i>See also transformation, matrix</i>
54	composition	
55	behavior	270
56	examples	271–73
57	optimization	270
58	rules	269
59	coordinate rounding	255
60	coordinate space	174
61	composability	54
62	effective coordinate space	9
63	described	46
64	mentioned	53, 63, 92, 103, 117, 127, 143, 149, 162, 168
65	mentioned	49
67	origin	255
68	transformation	<i>See transformation</i>
69	units	255
70	x-axis	255
71	y-axis	255
72	degenerate segments	283
73	implementation limits	<i>See implementation limits</i>
74	page dimensions	<i>See page</i>
75	pixel	
76	center location	<i>See layout, pixel</i>
77	inclusion	256
78	placement	256
79	placement behavior for glyphs	257
80	placement error maximum	257
81	rendering	256
82	sub-pixel masking	257
83	PrintTicket interactions	55
84	shape abutment	257
85	line	
86	characteristics of	<i>See stroke</i>
87	curved	<i>See geometry, segment</i>
88	drawing of	<i>See path</i>
89	geometry of	<i>See geometry, segment, line</i>
90	linear gradient	<i>See brush, linear gradient brush</i>
91	link	<i>See hyperlink</i>
92	link target	<i>See hyperlink, target</i>
93	list	<i>See document structure, list</i>

---

**M**

95	markup compatibility	
96	digital signature, impacted by	248
97	document structure, usage in	231
98	mentioned	1
99	namespace	<i>See namespace</i>
100	mentioned	45
101	preprocessing requirements	45
102	processing required	44
103	property elements, usage with	46
104	resource dictionary, usage in	170
105	usage of	44



1 memory management, device .....*See* interleaving; discard  
 2 control  
 3 miter ..... *See* stroke, line, join

---

4 **N**

5 name  
 6 elements applied to .....161  
 7 link target correspondence .....233  
 8 link target, specifying as .....51  
 9 purpose of .....232  
 10 resource entries, prohibited for .....233  
 11 syntax .....233  
 12 uniqueness .....232  
 13 named color.....*See* color, color space, named color  
 14 namespace.....393  
 15 content type .....393  
 16 core properties.....393  
 17 digital signatures .....393  
 18 discard control .....393  
 19 document .....393  
 20 document sequence.....393  
 21 document structure .....393  
 22 markup compatibility .....393  
 23 page.....393  
 24 relationships.....393  
 25 resource dictionary key .....393  
 26 signature definitions .....393  
 27 story fragments .....393  
 28 naming of parts.....*See* parts, naming recommendations  
 29 natural language .....*See* language  
 30 n-channel color ..... *See* color, color space, n-channel

---

31 **O**

32 opacity  
 33 blending ..... *See* color, blending  
 34 brush initial opacity .....127  
 35 composition effects .....270  
 36 computations .....266–68  
 37 described .....162  
 38 elements applied to .....161  
 39 mask ..... *See* opacity mask  
 40 of canvas .....56  
 41 of color .....33, 118, 143, 149, 195, 198, 199, 200, 201, 202,  
 42 209, 210  
 43 of glyphs .....92  
 44 of image brush .....120  
 45 of linear gradient brush.....143  
 46 of path .....63  
 47 of pixel formats .....203, 204, 205  
 48 of radial gradient brush.....149  
 49 of solid color brush.....118  
 50 of stroke .....270  
 51 of visual brush .....123  
 52 pre-multiplied alpha .....268

53 superluminous colors.....268  
 54 transparent brush tiling .....286  
 55 value range .....266  
 56 opacity mask  
 57 brush, filling with ..... *See* brush  
 58 described .....157  
 59 elements applied to .....161, 162  
 60 example .....157, 158  
 61 of canvas.....56, 190  
 62 of glyphs.....92, 192  
 63 of path .....63, 191  
 64 Open Packaging Conventions  
 65 ordering  
 66 simple ..... **10**  
 67 Open Packaging Conventions  
 68 ordering  
 69 interleaved ..... **10**  
 70 Open Packaging Conventions  
 71 package ..... **10**  
 72 Open Packaging Conventions  
 73 package  
 74 packaging model ..... **10**  
 75 Open Packaging Conventions  
 76 package  
 77 relationship ..... **10**  
 78 Open Packaging Conventions  
 79 physical model ..... **10**  
 80 Open Packaging Conventions  
 81 interleaving  
 82 piece ..... **11**  
 83 Open Packaging Conventions  
 84 physical model .....21  
 85 Open Packaging Conventions  
 86 package  
 87 packaging model .....21  
 88 Open Packaging Conventions  
 89 interleaving  
 90 optimization .....237  
 91 Open Packaging Conventions  
 92 interleaving  
 93 parsing head-first vs. tail first .....238  
 94 Open Packaging Conventions  
 95 interleaving  
 96 consumer considerations .....243  
 97 Open Packaging Conventions  
 98 digital signature .....*See* digital signature  
 99 Open Packaging Conventions  
 100 core properties .....253  
 101 Open Packaging Conventions  
 102 core properties  
 103 namespace ..... *See* namespace  
 104 Open XML Paper Specification  
 105 document format..... *See* OpenXPS Document format  
 106 organization of .....19  
 107 OpenType font ..... *See* font:OpenType  
 108 OpenXPS Document format ..... **10**, 19, 23  
 109 content types .....394

1	described .....	1, 49	58	<Intent> .....	252, 305
2	example .....	25	59	<SignatureDefinition> .....	250, 327
3	extensibility .....	See markup compatibility	60	<SignatureDefinitions> .....	250, 328
4	fixed payload .....	9, 23	61	<SignBy> .....	253, 327
5	fixed payload root .....	10, 23	62	<SigningLocation> .....	253, 328
6	illustrated .....	21	63	<SpotLocation> .....	251, 329
7	language .....	See language	64	discard control	
8	markup elements .....	See OpenXPS elements	65	<Discard> .....	244, 291
9	parts .....	21, 23	66	<DiscardControl> .....	244, 292
10	payload .....	23	67	document structure	
11	properties		68	<DocumentOutline> .....	213, 292
12	attribute syntax .....	46	69	<DocumentStructure.Outline> .....	213, 293
13	composability .....	46, 161	70	<DocumentStructure> .....	212, 293
14	described .....	45	71	<FigureStructure> .....	230, 294
15	element syntax .....	46	72	<ListItemStructure> .....	230, 308
16	model .....	45	73	<ListStructure> .....	229, 309
17	ordering .....	46, 407	74	<NamedElement> .....	231, 309
18	property attribute .....	11	75	<OutlineEntry> .....	213, 310
19	property element .....	11	76	<ParagraphStructure> .....	227, 312
20	property value .....	11, 45	77	<SectionStructure> .....	227, 326
21	property .....	11	78	<Story> .....	215, 329
22	relationship types .....	395	79	<StoryBreak> .....	227, 330
23	relationships .....	21, 26	80	<StoryFragment> .....	223, 330
24	RELAX NG schema		81	<StoryFragmentReference> .....	216, 332
25	OpenXPS Document .....	379	82	<StoryFragments> .....	222, 331
26	RELAX NG schema		83	<TableCellStructure> .....	229, 332
27	signature definitions .....	377	84	<TableRowGroupStructure> .....	228, 333
28	RELAX NG schema		85	<TableRowStructure> .....	228, 333
29	resource dictionary key .....	381	86	<TableStructure> .....	228, 334
30	RELAX NG schema		87	document-level	
31	document structure .....	383	88	<DocumentReference> .....	49, 292
32	RELAX NG schema		89	<FixedDocument> .....	50, 294
33	discard control .....	385	90	<FixedDocumentSequence> .....	49, 294
34	RELAX NG schema		91	<LinkTarget> .....	52, 308
35	3D-Graphic Content .....	386	92	<PageContent.LinkTargets> .....	51, 311
36	root .....	49	93	<PageContent> .....	50, 311
37	versioning .....	See markup compatibility	94	page-level	
38	XML		95	<ArcSegment> .....	75, 287
39	DTDs prohibited .....	44	96	<Brush3D> .....	457
40	markup design .....	43	97	<Canvas.Clip> .....	170, 290
41	markup model .....	45–47	98	<Canvas.OpacityMask> .....	190, 290
42	namespaces .....	45, 393	99	<Canvas.RenderTransform> .....	178, 291
43	Unicode encodings permitted .....	44	100	<Canvas.Resources> .....	164, 291
44	usage .....	44	101	<Canvas> .....	56, 288
45	whitespace .....	47	102	<FixedPage.Resources> .....	163, 296
46	XML and XSI namespace usage .....	45	103	<FixedPage> .....	53, 294
47	XML schema (XSD)		104	<Glyphs.Clip> .....	173, 301
48	3D-Graphic Content .....	374	105	<Glyphs.Fill> .....	115, 301
49	characteristics .....	45	106	<Glyphs.OpacityMask> .....	192, 301
50	discard control .....	373	107	<Glyphs.RenderTransform> .....	180, 302
51	document structure .....	367	108	<Glyphs> .....	92, 296
52	OpenXPS Document .....	341	109	<GradientStop> .....	156, 302
53	resource dictionary key .....	365	110	<ImageBrush.Transform> .....	182, 304
54	signature definitions .....	339	111	<ImageBrush> .....	119, 303
55	validity requirement .....	45	112	<LinearGradientBrush.GradientStops> .....	148, 307
56	OpenXPS elements		113	<LinearGradientBrush.Transform> .....	186, 307
57	digital signature		114	<LinearGradientBrush> .....	143, 305

1	<MatrixTransform>.....	174, 309
2	<Path.Clip>.....	172, 317
3	<Path.Data>.....	66, 317
4	<Path.Fill>.....	67, 317
5	<Path.OpacityMask>.....	191, 317
6	<Path.RenderTransform>.....	179, 318
7	<Path.Stroke>.....	68, 318
8	<Path>.....	62, 312
9	<PathFigure>.....	74, 319
10	<PathGeometry.Transform>.....	181, 320
11	<PathGeometry>.....	71, 319
12	<PolyBezierSegment>.....	79, 321
13	<PolyLineSegment>.....	80, 321
14	<PolyQuadraticBezierSegment>.....	81, 322
15	<RadialGradientBrush.GradientStops>.....	155, 325
16	<RadialGradientBrush.Transform>.....	187, 325
17	<RadialGradientBrush>.....	149, 322
18	<ResourceDictionary>.....	165, 325
19	<SolidColorBrush>.....	118, 328
20	<VisualBrush.Transform>.....	183, 336
21	<VisualBrush.Visual>.....	124, 337
22	<VisualBrush>.....	123, 334
23	optimization	
24	for streaming consumption.....	See Open Packaging
25	Conventions, interleaving	
26	of composition rules.....	270
27	of digital signatures.....	245
28	of glyphs.....	112
29	of interleaving.....	239–43
30	of named element location.....	212
31	of pixel placement rules.....	257
32	outline.....	See document structure, outline
<hr/>		
33	<b>P</b>	
34	packaging model.....	See Open Packaging Conventions
35	page	
36	bleed area.....	53, 54
37	content area.....	53, 55, 255
38	height of.....	53, 255
39	height, advisory.....	51
40	layout.....	See also layout
41	markup grouping.....	See canvas
42	namespace.....	See namespace
43	order of pages.....	50
44	orientation of.....	55
45	reference to.....	50
46	root of.....	53
47	scaling for print.....	55
48	uniqueness of.....	51
49	width of.....	53, 255
50	width, advisory.....	51
51	paragraph.....	See document structure, paragraph
52	part.....	10
53	part name.....	10
54	PrintTicket.....	40
55	parts.....	See also OpenXPS Document format
56	core properties.....	See Open Packaging Conventions, Standard
57	digital signature	
58	certificate	
59	part.....	See Open Packaging Conventions, Standard
60	digital signature origin.....	See Open Packaging Conventions,
61	Standard	
62	DiscardControl.....	24, 244
63	DocumentStructure.....	24, 40, 211
64	FixedDocument.....	23, 27
65	FixedDocumentSequence.....	23, 27
66	FixedPage.....	23, 28
67	font.....	23, 35–39
68	ICC profile.....	24
69	image.....	24, 28–34
70	naming recommendations.....	41–43
71	PrintTicket.....	24
72	remote resource dictionary.....	24, 39
73	SignatureDefinitions.....	24, 40
74	StoryFragments.....	24, 41, 217
75	thumbnail.....	24, 34
76	thumbnail, package.....	See Open Packaging Conventions,
77	Standard	
78	XML digital signature.....	See Open Packaging Conventions,
79	Standard	
80	path	
81	clipping.....	See clipping
82	described.....	61, 62
83	fill brush.....	63
84	geometry, reference to.....	63, 66
85	opacity.....	See opacity
86	opacity mask.....	See opacity mask
87	shape.....	See geometry
88	stroke brush.....	63
89	stroke control.....	See stroke
90	transformation.....	See transformation
91	usage described.....	66
92	payload.....	10, See OpenXPS Document format
93	physical imageable size.....	10
94	physical media size.....	10
95	physical model.....	See Open Packaging Conventions
96	physical organization.....	See Open Packaging Conventions,
97	interleaving; ZIP	
98	pixel.....	See layout, pixel
99	pixel snapping.....	See stroke, line
100	PNG.....	See image, PNG
101	positioning content.....	See layout; transformation, matrix
102	primary fixed payload root.....	11
103	printing	
104	bleed area.....	See page, bleed area
105	content area.....	See page, content area
106	device fonts.....	See font, device font
107	digital signature.....	See digital signature
108	discard control.....	See discard control
109	font, print and preview restricted..	See font, licensing rights
110	interleaving.....	See Open Packaging Conventions, interleaving
111	layout.....	See layout

1	orientation .....	See PrintTicket keywords
2	PrintTicket .....	See PrintTicket; PrintTicket keywords
3	resource constraints.....	See discard control
4	scaling .....	See PrintTicket keywords
5	PrintTicket.....	<b>11</b>
6	empty PrintTicket, markup of .....	239
7	mapping content levels to parts.....	40
8	parts .....	See parts
9	relationships.....	See relationships
10	producer .....	<b>11</b>
11	bleed size .....	<b>11</b>
12	content size .....	<b>11</b>
13	implementation burden .....	396–449
14	media size .....	<b>11</b>
15	property .....	See OpenXPS Document format, properties

**R**

17	radial gradient .....	See brush, radial gradient brush
18	raster graphics .....	See image
19	reading order .....	See accessibility, reading order
20	relationship.....	<b>11</b>
21	StartPart.....	<b>10</b>
22	relationships .....	See also OpenXPS Document format
23	core properties.....	26
24	digital signature.....	26
25	digital signature certificate .....	26
26	digital signature definitions.....	26
27	digital signature origin .....	26
28	DiscardControl .....	26
29	DocumentStructure.....	26
30	external prohibited .....	26
31	namespace .....	See namespace
32	PrintTicket .....	26
33	purpose of .....	26
34	relationship types, reference table .....	395
35	required resource.....	26
36	restricted font .....	27, 36, 38
37	StartPart.....	23, 27
38	StoryFragments.....	27
39	thumbnail.....	27
40	usage of.....	27
41	relationships part.....	<b>11</b>
42	rendering .....	See also layout
43	rules described.....	255–86
44	required part .....	<b>11</b>
45	resource.....	See resource dictionary, resource definition
46	resource constraints .....	See discard control
47	resource definition .....	See resource dictionary, resource definition
48	resource dictionary.....	<b>11</b>
49	described.....	162, <b>165</b>
50	example.....	163, 164, 166, 167, 168
51	markup compatibility usage.....	170
52	remote ..	<b>11</b> , See resource part, remote resource dictionary
53	resource definition.....	<b>11</b>

55	described.....	162, 165
56	key	
57	described .....	162
58	namespace.....	See namespace
59	on canvas.....	56
60	on geometry .....	71
61	on glyphs.....	92
62	on image brush .....	120
63	on linear gradient brush .....	143
64	on matrix transformation .....	174
65	on path .....	63
66	on radial gradient brush .....	149
67	on solid color brush .....	118
68	on visual brush.....	123
69	uniqueness.....	169
70	language.....	166
71	locating.....	169
72	namespace prefixes, interpreting .....	166
73	referencing previously-defined resources... ..	166, 167, 169
74	usefulness of path, glyphs, and canvas as .....	163
75	sharing .....	See resource part, remote resource dictionary
76	specifying.....	162, 163–67
77	where usable .....	163
78	resource part	
79	color profile .....	See color, color profile
80	font .....	See font
81	image .....	See image
82	reference to .....	26, 28
83	relationships to.....	See relationships
84	remote resource dictionary .....	24, 26, 163, <b>167</b>
85	required resource .....	26
86	usage of .....	26
87	resource reference .....	<b>11</b>
88	described .....	162, 168
89	example .....	168
90	scope.....	169
91	syntax.....	168
92	rotating content .....	See transformation, matrix

**S**

94	scalable vector graphics (SVG) .....	See geometry, abbreviated
95	syntax	
96	scaling	
97	content .....	See transformation, matrix
98	for print.....	See page:scaling for print
99	scRGB .....	See color, color space, scRGB
100	search.....	See find
101	section.....	See document structure, section
102	segment .....	See geometry, segment
103	selection	
104	behavior.....	234
105	caret stop.....	92, 111
106	document structure, enabled by .....	211, 234
107	fragment type, behavior depending on .....	234
108	mentioned .....	104, 111, 211

1	order.....	234	53	<b>T</b>	
2	recommended.....	234			
3	shear.....	<i>See transformation, matrix</i>	54	table.....	<i>See document structure, table, See document structure,</i>
4	signature definitions.....	<i>See digital signature</i>	55	table	
5	signature spot .. <b>12</b> , ..	<i>See digital signature, signature definitions</i>	56	table of contents.....	<i>See document structure, outline</i>
6	skewing content.....	<i>See transformation, matrix</i>	57	text	
7	sRGB.....	<i>See color, color space, sRGB</i>	58	baseline.....	<i>See text, glyph, baseline</i>
8	starting part.....	<i>See relationships, StartPart</i>	59	bidirectional.....	92, 108
9	story.....	<i>See document structure, story</i>	60	bold.....	<i>See text, style simulation</i>
10	story fragment.....	<i>See document structure, story fragment</i>	61	clipping.....	<i>See clipping</i>
11	stream.....	<b>12</b>	62	fill brush.....	92, 115
12	stretching content.....	<i>See transformation, matrix</i>	63	font.....	<i>See also font</i>
13	stroke		64	device font, reference to.....	92, 111
14	brush.....	<i>See path, stroke brush</i>	65	reference to.....	92
15	consistent nominal width.....	<i>See stroke, hairline</i>	66	glyph	
16	contour rendering.....	274	67	advance width.....	97, 98, 102, 103
17	dash		68	baseline.....	97
18	cap.....	275	69	black box.....	97
19	flat.....	275	70	cluster map.....	98–102, 102, <b>103</b> , 104, 421
20	round.....	276	71	indices	
21	square.....	275	72	described.....	102
22	style.....	63	73	reference to non-existent glyph.....	102
23	triangle.....	276	74	restriction of length.....	102
24	offset.....	63	75	specifying.....	92
25	overlapping.....	277	76	syntax.....	102
26	style.....	63	77	metrics.....	97
27	drawing algorithm, symmetry of.....	274	78	offset.....	98, 102, 103, 107, 421
28	edge parallelization.....	274	79	origin.....	97, 98
29	fill rule, independence from.....	284	80	origin, sideways.....	97
30	hairline.....	284	81	side-bearing, bottom.....	98
31	line		82	side-bearing, left.....	97
32	cap.....	277	83	side-bearing, right.....	97
33	end.....	63	84	side-bearing, top.....	98
34	flat.....	278	85	glyphs usage for.....	91
35	for dashed stroke.....	278	86	italic.....	<i>See text, style simulation</i>
36	round.....	278	87	markup.....	92
37	square.....	278	88	markup examples.....	112
38	start.....	63	89	markup optimization.....	112
39	triangle.....	278	90	opacity.....	<i>See opacity</i>
40	join		91	opacity mask.....	<i>See opacity mask</i>
41	bevel.....	280	92	position.....	<i>See also text, glyph, offset</i>
42	miter.....	281–83	93	horizontal text.....	92
43	round.....	279	94	vertical text.....	105
44	style.....	63	95	sideways (vertical).....	92
45	miter limit.....	63	96	advance width.....	102, <b>106</b>
46	pixel snapping.....	63	97	bidirectional text, intersection with.....	107
47	thin stroke anti-aliasing behavior.....	257	98	described.....	105
48	zero-length, avoided.....	75	99	example, sideways.....	108
49	multi-figure path, behavior with.....	284	100	example, vertical.....	109
50	phase control.....	274	101	horizontal text, including.....	106
51	segments, mixed stroked and non-stroked.....	284	102	origin calculation.....	106
52	thickness.....	63	103	vertical glyphs, preference for.....	106
			104	size.....	92
			105	style simulation.....	92, <b>105</b>
			106	transformation.....	<i>See transformation</i>
			107	underline.....	<i>See path</i>
			108	Unicode string.....	<i>See also text, glyph</i>

1 escaping open brace character .....104

2 mapping code units to glyphs .....98

3 normalization prohibited .....104

4 specifying .....92

5 Unicode control marks, inclusion of .....104

6 Unicode scalar value, split into code units .....98

7 unmappable code unit behavior .....104

8 usage of .....**104**

9 UTF-16 code units, consisting of .....98

10 vertical ..... *See text, sideways (vertical)*

11 thread ..... *See document structure, story*

12 thumbnail .....**12**

13 described ..... **24, 34**

14 formats ..... *See image*

15 parts ..... *See parts*

16 relationship .....27

17 usage .....34

18 TIFF ..... *See image, TIFF*

19 transformation

20 composability .....256

21 described .....174, **256**

22 effective transform .....*See transformation, composability*

23 elements applied to .....161, 162

24 matrix

25 abbreviated syntax .....175

26 abbreviated syntax example .....178

27 described .....174

28 example .....176

29 inverting x-axis .....175

30 inverting y-axis .....175

31 multiplying .....174

32 positioning .....176

33 rotating .....176

34 scaling .....175

35 skewing .....175

36 specifying .....174

37 mentioned .....255

38 non-invertible transform, rendering of elements with ...256

39 of brush .....117

40 of canvas .....56, 178

41 of geometry .....71, 181

42 of glyphs ..... 92, 180

43 of image brush ..... 120, 182

44 of linear gradient brush ..... 143, 186

45 of path ..... 63, 179

46 of radial gradient brush ..... 149, 187

47 of tiles ..... 183

48 of tiles, example ..... 185

49 of visual brush ..... 123, 183

50 transparency ..... *See opacity*

51 TrueType collection (TTC) font ... *See font:TrueType collection (TTC)*

52 (TTC)

53 TrueType font ..... *See font:TrueType*

---

54 **V**

55 vector graphics ..... *See path*

56 versioning ..... *See markup compatibility*

---

57 **W**

58 whitespace *See OpenXPS Document format, XML, whitespace*

59 Windows Color System (WCS) ..... *See color, color profile, Windows Color System (WCS) profile*

60 Windows Color System (WCS) profile

61 JPEG XR ..... *See image, JPEG XR*

---

62 **X**

63 XML ..... *See OpenXPS Document format, XML*

64 XML namespaces ..... *See namespace; OpenXPS Document format, XML, namespaces*

65 format, XML, namespaces

---

66 **Z**

67 ZIP

68 archive ..... **12, 21**

69 item ..... 12

70 utilities ..... 37