Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 7 May 2026

A. Brotman
Comcast, Inc
T. Adams
Proofpoint, Inc

3 November 2025

# SVG Tiny Portable/Secure
# draft-svg-tiny-ps-abrotman-11

## Abstract

This document specifies SVG Tiny Portable/Secure (SVG Tiny PS) -- A Scalable Vector Graphics (SVG) profile to be used with documents that are intended for use with more secure requirements, and in some cases, in conjunction with a limited rendering engine.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 7 May 2026.

Table of Contents



## 1. Introduction

Scalable Vector Graphics (SVG) is a standard that defines vector-based graphics for the web, graphics that can be expressed in XML format. SVG is an extensible format, with support for various metadata elements and animation. In some cases, it is preferable to have an SVG profile that is targeted at smaller platforms, and/or to have security in mind. This document defines a profile of SVG for such use cases, called the SVG Tiny PS profile, that limits the allowed elements to a subset of the features supported by the overall SVG specification.

The SVG standard is defined by the World Wide Web Consortium (W3C) and the current version is 1.1 [SVG]. The W3C also defines a simplified profile of SVG published as SVG Tiny [SVGT], with the current version at 1.2. This document defines the SVG Tiny PS profile in the terms of the simplified SVG Tiny 1.2 as the base. The SVG Tiny PS profile identifies elements that are required, disallowed, or otherwise allowed from the set of elements defined by SVG Tiny 1.2. No elements are added to SVG Tiny PS, and any elements not explicitly mentioned in this document follow the guidance provided by SVG Tiny 1.2.

## 2. SVG Tiny PS Profile

This document defines the SVG Tiny PS profile in the terms of the simplified SVG Tiny 1.2 as the base. The following sections reference sections within the SVG Tiny PS profile that are required, disallowed, or otherwise allowed from the set of elements defined by SVG Tiny 1.2. No elements are added to SVG Tiny PS that are not found within SVG 1.1 or SVG Tiny 1.2, and any elements not explicitly mentioned in this document follow the guidance provided by SVG Tiny 1.2.

### 2.1. Specific Instructions for the SVG Tiny PS Documents

The elements and attributes below are required within the root <svg> element of an SVG Tiny PS document.

* Document Definition: The SVG Tiny PS document MUST be defined as described in SVG Tiny 1.2 Section 1.3. This includes setting the following attributes as follows:

```
xmlns="http://www.w3.org/2000/svg"
version="1.2"
baseProfile="tiny-ps"
```

[ Note to WG: Based on other baseProfile, this should be the minimum profile required to render the image. Tiny PS should be a subset, but we’re unsure if we should create a smaller profile. ]

* title: The <title> element MUST be present and its contents MUST NOT be empty. The title element MUST occur only once as a sub-element of the svg element. The contents SHOULD be no more 64 characters.

* desc: The <desc> element MAY be present with a longer description. If present, its contents MUST NOT be empty, and care SHOULD be given to minimize the file size of the document.

### 2.2. Permitted Sections with Restrictions

The following sections from SVG Tiny 1.2 MAY be used within an SVG Tiny PS document with the further restrictions as noted.

* Section 8: Paths - This section defines methods used for defining vector-based paths that can be rendered. The attributes "d" and "pathLength" MUST NOT be animated. The elements described within "8.2.1 Animating path data" MUST NOT be present within an SVG Tiny PS document.

* Section 9: Basic Shapes - This section defines primitives that can be used to render basic shapes. The attributes "x", "y", "width", "height", "rx", and "ry" MUST NOT be animated.

* Section 10: Text - This sections describes the methods for defining text that should be rendered similar to any vector-based graphic elements. The "editable" attribute SHOULD NOT be present in an SVG Tiny PS document, if it is, it MUST be set to "none". The attributes "x", "y", and "rotate" MUST NOT be animated.

* Section 11: Painting - This section describes the methods used to render the paths ("stroked") and shapes ("filled") visible. Any of the attributes described MUST NOT be animated within an SVG Tiny PS document.

## 2.3. Disallowed Sections, Elements, and Attributes

The elements and attributes described in the sections below MUST NOT be present in an SVG Tiny PS document:

* Section 5.7: The ’image’ Element - This section defines methods for displaying a raster image within the context of an SVG image.

* Section 5.8.2: The ’switch’ Element - This section defines methods for conditional processing of the document.

* Section 12: Multimedia - This section describes the elements that augment an image with audio or video.

* Section 13: Interactivity - This section describes elements related to user actions and responsiveness to those events.

* Section 14: Linking - This section describes links to internal and external end points.

* Section 15: Scripting - This section describes methods for adding executable content.

* Section 16: Animation - This section describes methods for transforming or otherwise animating elements over time.

The attributes below are specifically called out from the sections above due to their widespread use. For a more complete list, refer to the full content in the sections above.

* zoomAndPan: The "zoomAndPan" attribute SHOULD NOT be present in an SVG Tiny PS document. If it is present, it MUST be set to "disable".

* externalResourcesRequired: The "externalResourcesRequired" attribute SHOULD NOT be present in an SVG Tiny PS document. If it is present, it MUST be set to "false".

* focusable: The "focusable" attribute SHOULD NOT be used within an SVG Tiny PS document. If it is present, it MUST be set to "false".

* snapshotTime: The "snapshotTime" attribute SHOULD NOT be used within an SVG Tiny PS document. If it is present, it MUST be set to "none".

* playbackOrder: The "playbackOrder" attribute SHOULD NOT be used within an SVG Tiny PS document. If it is present, it MUST be set to "all".

* timelineBegin: The "timelineBegin" attribute SHOULD NOT be used within an SVG Tiny PS document. If it is present, it MUST be set to "onLoad".

### 2.4. Additional Considerations for Creating SVG Tiny PS Documents

The following are special considerations that SHOULD be taken when creating an SVG Tiny PS document:

* The file size of SVG Tiny PS documents SHOULD be as small as possible, and SHOULD NOT exceed 32 kilobytes. That size should be evaluated when the document is uncompressed.

* An SVG Tiny PS document MUST include at least two colors when rendered.

* Section 17: Fonts - This sections describes the methods for defining specific characters to be used within an SVG document. Embedded character definitions MAY be used in an SVG Tiny PS document, but the document SHOULD only include the characters to be rendered, and SHOULD NOT include any additional characters that are not to be rendered.

* Section 18: Metadata - This section summarizes standard methods for including structured data about the content within an SVG document. While metadata can support utility and accessibility and MAY be included in an SVG Tiny PS document, careful consideration SHOULD be given to how the metadata element and extensible metadata attributes are used to avoid increasing the file size of the document.

* An SVG Tiny PS document MAY include transparent elements, however care SHOULD be taken to understand the implications of transparency when rendered. Given that the author has no control over how the logo is ultimately displayed, transparency to the underlying display mechanism could result in undesirable effects.

## 3. Security Considerations

Many of the requirements imposed by the SVG Tiny PS profile are designed to minimize potential security concerns introduced by interactive content into a user experience not typically designed to handle it. Every consideration has been made to reduce the scope of SVG images to the minimum elements necessary to render locally defined, static vector graphics.

### 3.1. Portability

The SVG documents meant to be used with this profile should be fairly portable, and should likely be self-contained. The document authors cannot be certain that the client will have an active network connection.

## 4. Contributors

TBD

## 5. Example SVG Tiny PS Document

```
<?xml version="1.0"?>
<svg width="400px" height="400px" xmlns="http://www.w3.org/2000/svg"
    version="1.2" baseProfile="tiny-ps"
    zoomAndPan="disable" externalResourcesRequired="false">
  <title>Example, Inc.</title>
  <desc>Logo for Example, Inc.</desc>
  <rect x="1" y="1" width="399" height="399" fill="teal"
     stroke="gray" stroke-width="9"/>
  <circle cx="200" cy="200" r="125" fill="white"
     stroke="black" stroke-width="2"/>
  <polyline fill="gray" stroke="silver" stroke-width="9"
     points="40,30 25,40 100,330 310,270 290,250 120,300 40,26"/>
</svg>
```

## 6. Notes

### 6.1. References:

https://www.oasis-open.org/committees/relax-ng/compact-20021121.html (https://www.oasis-open.org/committees/relax-ng/compact-20021121.html)

### 6.2. Known issues with various SVG editing software

* Illustrator ** Will add x/y attributes to the svg element. These are not permitted.

## 7. Validation RNC

The following RNC [RNC] schema can be used to validate a SVG Tiny PS document. The official version of this document resides at:

http://bimigroup.org/resources/SVG_PS-latest.rnc.txt (http://bimigroup.org/resources/SVG_PS-latest.rnc.txt)

And is the one that should be used if there is any conflict.

```
#---  SVG 1.2 PS RNC schema

default namespace = "http://www.w3.org/2000/svg"
#namespace ns1 = "http://www.w3.org/1999/xlink"

start = svg
svg =
  element svg {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | xsd:string }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { xsd:string }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { xsd:string }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
```

```
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
```

```
       attribute width { xsd:string }?,
       attribute height { xsd:string }?,
       attribute viewBox { text }?,
       attribute baseProfile { "tiny-ps" },
       attribute contentScriptType { xsd:string }?,
       attribute externalResourcesRequired { "false" }?,
       attribute focusable { "false" }?,
       attribute playbackOrder { "all" }?,
       attribute preserveAspectRatio { xsd:string { pattern = "\s*(none|xMidYMid)\s*(meet
)?\s*" } }?,
       attribute snapshotTime { "none" }?,
       attribute timelineBegin { "onLoad" }?,
       attribute version { "1.2" },
       attribute zoomAndPan { "disable" }?,
       attribute viewport-fill { "none" | xsd:string }?,
       attribute viewport-fill-opacity { "inherit" | xsd:string }?
       (svgTitle),
       (desc
        | path
        | rect
        | circle
        | line
        | ellipse
        | polyline
        | polygon
        | solidColor
        | textArea
        | linearGradient
        | radialGradient
        | \text
        | g
        | defs
        | metadata
        | use)*
     }

   desc =
     element desc {
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
```

```
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       ((attribute display {
           "inline"
           | "block"
           | "list-item"
           | "run-in"
           | "compact"
           | "marker"
           | "table"
           | "inline-table"
           | "table-row-group"
           | "table-header-group"
           | "table-footer-group"
           | "table-row"
           | "table-column-group"
           | "table-column"
           | "table-cell"
           | "table-caption"
           | "none"
           | "inherit"
         }?,
         attribute visibility { "visible" | "hidden" | "collapse"
           | "inherit" }?,
         attribute image-rendering {
           "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
         }?,
         attribute shape-rendering {
           "auto"
           | "optimizeSpeed"
           | "crispEdges"
           | "geometricPrecision"
           | "inherit"
         }?,
         attribute text-rendering {
           "auto"
           | "optimizeSpeed"
           | "optimizeLegibility"
           | "geometricPrecision"
           | "inherit"
         }?,
         attribute buffered-rendering {
           "auto" | "dynamic" | "static" | "inherit"
         }?)
```

```
        & (attribute viewport-fill { "none" | xsd:string }?,
           attribute viewport-fill-opacity { "inherit" | xsd:string }?)),
      text
    }
  svgTitle =
    element title {
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute requiredFonts { xsd:string }?,
      attribute systemLanguage { xsd:string }?,
      ((attribute display {
          "inline"
          | "block"
          | "list-item"
          | "run-in"
          | "compact"
          | "marker"
          | "table"
          | "inline-table"
          | "table-row-group"
          | "table-header-group"
          | "table-footer-group"
          | "table-row"
          | "table-column-group"
          | "table-column"
          | "table-cell"
          | "table-caption"
          | "none"
          | "inherit"
        }?,
        attribute visibility { "visible" | "hidden"
          | "collapse" | "inherit" }?,
        attribute image-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?,
```

```
        attribute shape-rendering {
          "auto"
          | "optimizeSpeed"
          | "crispEdges"
          | "geometricPrecision"
          | "inherit"
        }?,
        attribute text-rendering {
          "auto"
          | "optimizeSpeed"
          | "optimizeLegibility"
          | "geometricPrecision"
          | "inherit"
        }?,
        attribute buffered-rendering {
          "auto" | "dynamic" | "static" | "inherit"
        }?)
       & (attribute viewport-fill { "none" | xsd:string }?,
          attribute viewport-fill-opacity { "inherit" | xsd:string }?)),
      text
    }
  path =
    element path {
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute transform { xsd:string | "none" }?,
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
```

```
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
      attribute requiredFonts { xsd:string }?,
      attribute systemLanguage { xsd:string }?,
      attribute d { xsd:string }?,
      attribute pathLength { xsd:string }?
```

```
    }
  rect =
    element rect {
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute transform { xsd:string | "none" }?,
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
       & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
       & (attribute solid-color { xsd:string }?,
          attribute solid-opacity { "inherit" | xsd:string }?)
       & (attribute display-align {
```

```
            "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
       & (attribute stop-color { xsd:string }?,
          attribute stop-opacity { "inherit" | xsd:string }?)
       & (attribute font-family { "inherit" | xsd:string }?,
          attribute font-size { "inherit" | xsd:string }?,
          attribute font-style {
            "normal" | "italic" | "oblique" | "inherit"
          }?,
          attribute font-variant { "normal" | "small-caps" | "inherit" }?,
          attribute font-weight {
            "normal"
            | "bold"
            | "bolder"
            | "lighter"
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
      attribute requiredFonts { xsd:string }?,
      attribute systemLanguage { xsd:string }?,
      attribute x { xsd:string }?,
      attribute y { xsd:string }?,
      attribute width { xsd:string }?,
      attribute height { xsd:string }?,
      attribute rx { xsd:string }?,
      attribute ry { xsd:string }?
    }
  circle =
    element circle {
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
```

```
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute transform { xsd:string | "none" }?,
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
             "butt" | "round" | "square" | "inherit"
           }?,
           attribute stroke-linejoin {
             "miter" | "round" | "bevel" | "inherit"
           }?,
           attribute stroke-miterlimit { "inherit" | xsd:string }?,
           attribute stroke-width { "inherit" | xsd:string }?,
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
```

```
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
        attribute requiredFonts { xsd:string }?,
        attribute systemLanguage { xsd:string }?,
        attribute cx { xsd:string }?,
        attribute cy { xsd:string }?,
        attribute r { xsd:string }?
      }
    line =
      element line {
        (attribute id { xsd:NCName }
         | attribute xml:id { xsd:NCName })?,
        attribute xml:base { xsd:anyURI | xsd:string }?,
        attribute xml:lang { xsd:language? }?,
        attribute class { xsd:NMTOKENS }?,
        attribute role { xsd:string }?,
        attribute rel { xsd:string }?,
        attribute rev { xsd:string }?,
        attribute typeof { xsd:string }?,
        attribute content { xsd:string }?,
        attribute datatype { xsd:string }?,
        attribute resource { xsd:string }?,
        attribute about { xsd:string }?,
        attribute property { xsd:string }?,
        attribute xml:space { "default" | "preserve" }?,
        attribute transform { xsd:string | "none" }?,
        ((attribute fill-opacity { "inherit" | xsd:string }?,
          attribute stroke-opacity { "inherit" | xsd:string }?)
         & (attribute fill { "none" | xsd:string }?,
            attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
            attribute stroke { xsd:string }?,
            attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
            attribute stroke-dashoffset { "inherit" | xsd:string }?,
            attribute stroke-linecap {
              "butt" | "round" | "square" | "inherit"
            }?,
            attribute stroke-linejoin {
              "miter" | "round" | "bevel" | "inherit"
            }?,
            attribute stroke-miterlimit { "inherit" | xsd:string }?,
            attribute stroke-width { "inherit" | xsd:string }?,
            attribute color { xsd:string }?,
```

```
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
         & attribute vector-effect {
             "none" | "non-scaling-stroke" | "inherit"
           }?
         & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
         & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
         & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
         & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
         & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       attribute x1 { xsd:string }?,
       attribute y1 { xsd:string }?,
       attribute x2 { xsd:string }?,
       attribute y2 { xsd:string }?
     }
   ellipse =
     element ellipse {
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
```

```
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute transform { xsd:string | "none" }?,
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
             "butt" | "round" | "square" | "inherit"
           }?,
           attribute stroke-linejoin {
             "miter" | "round" | "bevel" | "inherit"
           }?,
           attribute stroke-miterlimit { "inherit" | xsd:string }?,
           attribute stroke-width { "inherit" | xsd:string }?,
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
```

```
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       attribute rx { xsd:string }?,
       attribute ry { xsd:string }?,
       attribute cx { xsd:string }?,
       attribute cy { xsd:string }?
     }
   polyline =
     element polyline {
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute transform { xsd:string | "none" }?,
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
```

```
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
             "butt" | "round" | "square" | "inherit"
           }?,
           attribute stroke-linejoin {
             "miter" | "round" | "bevel" | "inherit"
           }?,
           attribute stroke-miterlimit { "inherit" | xsd:string }?,
           attribute stroke-width { "inherit" | xsd:string }?,
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
         & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
         & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
         & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
         & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
         & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
         & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
```

```
          }?)),
        attribute requiredFonts { xsd:string }?,
        attribute systemLanguage { xsd:string }?,
        attribute points { xsd:string }?
      }
    polygon =
      element polygon {
        (attribute id { xsd:NCName }
         | attribute xml:id { xsd:NCName })?,
        attribute xml:base { xsd:anyURI | xsd:string }?,
        attribute xml:lang { xsd:language? }?,
        attribute class { xsd:NMTOKENS }?,
        attribute role { xsd:string }?,
        attribute rel { xsd:string }?,
        attribute rev { xsd:string }?,
        attribute typeof { xsd:string }?,
        attribute content { xsd:string }?,
        attribute datatype { xsd:string }?,
        attribute resource { xsd:string }?,
        attribute about { xsd:string }?,
        attribute property { xsd:string }?,
        attribute xml:space { "default" | "preserve" }?,
        attribute transform { xsd:string | "none" }?,
        ((attribute fill-opacity { "inherit" | xsd:string }?,
          attribute stroke-opacity { "inherit" | xsd:string }?)
         & (attribute fill { "none" | xsd:string }?,
            attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
            attribute stroke { xsd:string }?,
            attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
            attribute stroke-dashoffset { "inherit" | xsd:string }?,
            attribute stroke-linecap {
              "butt" | "round" | "square" | "inherit"
            }?,
            attribute stroke-linejoin {
              "miter" | "round" | "bevel" | "inherit"
            }?,
            attribute stroke-miterlimit { "inherit" | xsd:string }?,
            attribute stroke-width { "inherit" | xsd:string }?,
            attribute color { xsd:string }?,
            attribute color-rendering {
              "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
            }?)
         & attribute vector-effect {
             "none" | "non-scaling-stroke" | "inherit"
           }?
         & (attribute direction { "ltr" | "rtl" | "inherit" }?,
            attribute unicode-bidi {
              "normal" | "embed" | "bidi-override" | "inherit"
```

```
          }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       attribute points { xsd:string }?
     }
   solidColor =
     element solidColor {
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
             "butt" | "round" | "square" | "inherit"
           }?,
           attribute stroke-linejoin {
             "miter" | "round" | "bevel" | "inherit"
           }?,
           attribute stroke-miterlimit { "inherit" | xsd:string }?,
```

```
           attribute stroke-width { "inherit" | xsd:string }?,
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
         & attribute vector-effect {
             "none" | "non-scaling-stroke" | "inherit"
           }?
         & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
         & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
         & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
         & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
         & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
        (attribute id { xsd:NCName }
         | attribute xml:id { xsd:NCName })?,
        attribute xml:base { xsd:anyURI | xsd:string }?,
        attribute xml:lang { xsd:language? }?,
        attribute class { xsd:NMTOKENS }?,
        attribute role { xsd:string }?,
        attribute rel { xsd:string }?,
        attribute rev { xsd:string }?,
        attribute typeof { xsd:string }?,
```

```
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?
    }
  textArea =
    element textArea {
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
       & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
       & (attribute solid-color { xsd:string }?,
          attribute solid-opacity { "inherit" | xsd:string }?)
       & (attribute display-align {
            "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
       & (attribute stop-color { xsd:string }?,
          attribute stop-opacity { "inherit" | xsd:string }?)
       & (attribute font-family { "inherit" | xsd:string }?,
          attribute font-size { "inherit" | xsd:string }?,
          attribute font-style {
            "normal" | "italic" | "oblique" | "inherit"
          }?,
```

```
          attribute font-variant { "normal" | "small-caps" | "inherit" }?,
          attribute font-weight {
            "normal"
            | "bold"
            | "bolder"
            | "lighter"
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       attribute transform { xsd:string | "none" }?,
       attribute x { xsd:string }?,
       attribute y { xsd:string }?,
       attribute width { xsd:string | "auto" }?,
       attribute height { xsd:string | "auto" }?,
       (text)+
     }
   linearGradient =
     element linearGradient {
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
```

```
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
       & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
       & (attribute solid-color { xsd:string }?,
          attribute solid-opacity { "inherit" | xsd:string }?)
       & (attribute display-align {
            "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
       & (attribute stop-color { xsd:string }?,
          attribute stop-opacity { "inherit" | xsd:string }?)
       & (attribute font-family { "inherit" | xsd:string }?,
          attribute font-size { "inherit" | xsd:string }?,
          attribute font-style {
            "normal" | "italic" | "oblique" | "inherit"
          }?,
          attribute font-variant { "normal" | "small-caps" | "inherit" }?,
          attribute font-weight {
            "normal"
            | "bold"
            | "bolder"
            | "lighter"
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
      attribute gradientUnits { "userSpaceOnUse" | "objectBoundingBox" }?,
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
```

```
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute x1 { xsd:string }?,
      attribute y1 { xsd:string }?,
      attribute x2 { xsd:string }?,
      attribute y2 { xsd:string }?,
      (stop)*
    }
  radialGradient =
    element radialGradient {
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
       & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
       & (attribute solid-color { xsd:string }?,
```

```
          attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
          attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
          attribute font-size { "inherit" | xsd:string }?,
          attribute font-style {
            "normal" | "italic" | "oblique" | "inherit"
          }?,
          attribute font-variant { "normal" | "small-caps" | "inherit" }?,
          attribute font-weight {
            "normal"
            | "bold"
            | "bolder"
            | "lighter"
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
      attribute gradientUnits { "userSpaceOnUse" | "objectBoundingBox" }?,
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute cx { xsd:string }?,
      attribute cy { xsd:string }?,
      attribute r { xsd:string }?,
      (stop)*
    }
```

```
   \text =
     element text {
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "none" | xsd:string }?,
           attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
           attribute stroke { xsd:string }?,
           attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
           attribute stroke-dashoffset { "inherit" | xsd:string }?,
           attribute stroke-linecap {
             "butt" | "round" | "square" | "inherit"
           }?,
           attribute stroke-linejoin {
             "miter" | "round" | "bevel" | "inherit"
           }?,
           attribute stroke-miterlimit { "inherit" | xsd:string }?,
           attribute stroke-width { "inherit" | xsd:string }?,
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
```

```
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
            "start" | "center" | "end" | "inherit"
          }?)),
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute editable { "none" }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute requiredFonts { xsd:string }?,
      attribute systemLanguage { xsd:string }?,
      attribute transform { xsd:string | "none" }?,
      attribute x { xsd:string }?,
      attribute y { xsd:string }?,
      attribute rotate { xsd:string }?,
      (text)+
    }
  g =
    element g {
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
```

```
           attribute color { xsd:string }?,
           attribute color-rendering {
             "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
           }?)
         & attribute vector-effect {
             "none" | "non-scaling-stroke" | "inherit"
           }?
         & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
         & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
         & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
         & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
         & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
        (attribute id { xsd:NCName }
         | attribute xml:id { xsd:NCName })?,
        attribute xml:base { xsd:anyURI | xsd:string }?,
        attribute xml:lang { xsd:language? }?,
        attribute class { xsd:NMTOKENS }?,
        attribute role { xsd:string }?,
        attribute rel { xsd:string }?,
        attribute rev { xsd:string }?,
        attribute typeof { xsd:string }?,
        attribute content { xsd:string }?,
```

```
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute requiredFonts { xsd:string }?,
      attribute systemLanguage { xsd:string }?,
      attribute transform { xsd:string | "none" }?,
      (path
       | rect
       | circle
       | line
       | ellipse
       | polyline
       | polygon
       | solidColor
       | textArea
       | linearGradient
       | radialGradient
       | \text
       | g
       | defs
       | use)*
    }
  defs =
    element defs {
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
```

```
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
           attribute unicode-bidi {
             "normal" | "embed" | "bidi-override" | "inherit"
           }?)
        & (attribute solid-color { xsd:string }?,
           attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
             "auto" | "before" | "center" | "after" | "inherit"
           }?,
           attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       (path
        | rect
```

```
        | circle
        | line
        | ellipse
        | polyline
        | polygon
        | solidColor
        | textArea
        | linearGradient
        | radialGradient
        | \text
        | g
        | defs
        | use)*
    }
  use =
    element use {
      ((attribute fill-opacity { "inherit" | xsd:string }?,
        attribute stroke-opacity { "inherit" | xsd:string }?)
       & (attribute fill { "none" | xsd:string }?,
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
       & attribute vector-effect {
           "none" | "non-scaling-stroke" | "inherit"
         }?
       & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
       & (attribute solid-color { xsd:string }?,
          attribute solid-opacity { "inherit" | xsd:string }?)
       & (attribute display-align {
            "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
```

```
        & (attribute stop-color { xsd:string }?,
           attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
           attribute font-size { "inherit" | xsd:string }?,
           attribute font-style {
             "normal" | "italic" | "oblique" | "inherit"
           }?,
           attribute font-variant { "normal" | "small-caps" | "inherit" }?,
           attribute font-weight {
             "normal"
             | "bold"
             | "bolder"
             | "lighter"
             | "inherit"
           }?,
           attribute text-anchor {
             "start" | "middle" | "end" | "inherit"
           }?,
           attribute text-align {
             "start" | "center" | "end" | "inherit"
           }?)),
       (attribute id { xsd:NCName }
        | attribute xml:id { xsd:NCName })?,
       attribute xml:base { xsd:anyURI | xsd:string }?,
       attribute xml:lang { xsd:language? }?,
       attribute class { xsd:NMTOKENS }?,
       attribute role { xsd:string }?,
       attribute rel { xsd:string }?,
       attribute rev { xsd:string }?,
       attribute typeof { xsd:string }?,
       attribute content { xsd:string }?,
       attribute datatype { xsd:string }?,
       attribute resource { xsd:string }?,
       attribute about { xsd:string }?,
       attribute property { xsd:string }?,
       attribute xml:space { "default" | "preserve" }?,
       attribute requiredFonts { xsd:string }?,
       attribute systemLanguage { xsd:string }?,
       attribute transform { xsd:string | "none" }?,
       attribute href { xsd:string }?,
       attribute x { xsd:string }?,
       attribute y { xsd:string }?
     }
   stop =
     element stop {
       ((attribute fill-opacity { "inherit" | xsd:string }?,
         attribute stroke-opacity { "inherit" | xsd:string }?)
        & (attribute fill { "inherit" | xsd:string }?,
```

```
          attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
          attribute stroke { "inherit" | xsd:string }?,
          attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
          attribute stroke-dashoffset { "inherit" | xsd:string }?,
          attribute stroke-linecap {
            "butt" | "round" | "square" | "inherit"
          }?,
          attribute stroke-linejoin {
            "miter" | "round" | "bevel" | "inherit"
          }?,
          attribute stroke-miterlimit { "inherit" | xsd:string }?,
          attribute stroke-width { "inherit" | xsd:string }?,
          attribute color { xsd:string }?,
          attribute color-rendering {
            "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
          }?)
        & attribute vector-effect {
            "none" | "non-scaling-stroke" | "inherit"
          }?
        & (attribute direction { "ltr" | "rtl" | "inherit" }?,
          attribute unicode-bidi {
            "normal" | "embed" | "bidi-override" | "inherit"
          }?)
        & (attribute solid-color { "inherit" | xsd:string }?,
          attribute solid-opacity { "inherit" | xsd:string }?)
        & (attribute display-align {
            "auto" | "before" | "center" | "after" | "inherit"
          }?,
          attribute line-increment { "auto" | "inherit" | xsd:string }?)
        & (attribute stop-color { "inherit" | xsd:string }?,
          attribute stop-opacity { "inherit" | xsd:string }?)
        & (attribute font-family { "inherit" | xsd:string }?,
          attribute font-size { "inherit" | xsd:string }?,
          attribute font-style {
            "normal" | "italic" | "oblique" | "inherit"
          }?,
          attribute font-variant { "normal" | "small-caps" | "inherit" }?,
          attribute font-weight {
            "normal"
            | "bold"
            | "bolder"
            | "lighter"
            | "inherit"
          }?,
          attribute text-anchor {
            "start" | "middle" | "end" | "inherit"
          }?,
          attribute text-align {
```

```
            "start" | "center" | "end" | "inherit"
          }?)),
      (attribute id { xsd:NCName }
       | attribute xml:id { xsd:NCName })?,
      attribute xml:base { xsd:anyURI | xsd:string }?,
      attribute xml:lang { xsd:language? }?,
      attribute class { xsd:NMTOKENS }?,
      attribute role { xsd:string }?,
      attribute rel { xsd:string }?,
      attribute rev { xsd:string }?,
      attribute typeof { xsd:string }?,
      attribute content { xsd:string }?,
      attribute datatype { xsd:string }?,
      attribute resource { xsd:string }?,
      attribute about { xsd:string }?,
      attribute property { xsd:string }?,
      attribute xml:space { "default" | "preserve" }?,
      attribute offset { xsd:string }?
    }

  font =
    element font {
      attribute horiz-origin { xsd:string }?,
      attribute horiz-adv-x { xsd:string }?,
    (desc
    | glyph
    | font-face
    | hkern)*
    }

  glyph =
    element glyph {
      attribute unicode { xsd:string }?,
      attribute glyph-name { xsd:string }?,
      attribute arabic-form { xsd:string }?,
      attribute lang { xsd:string }?,
    (desc)*
    }


  hkern =
    element hkern {
      attribute u1 { xsd:string }?,
      attribute g1 { xsd:string }?,
      attribute u2 { xsd:string }?,
      attribute g2 { xsd:string }?,
      attribute k { xsd:string }?,
      (desc)*
```

```
      }

   font-face =
     element font-face {
       attribute font-family { xsd:string }?,
       attribute font-style { xsd:string }?,
       attribute font-weight { xsd:string }?,
       attribute font-variant { xsd:string }?,
       attribute font-stretch { xsd:string }?,
       attribute unicode-range { xsd:string }?,
       attribute panose-1 { xsd:string }?,
       attribute widths { xsd:string }?,
       attribute bbox { xsd:string }?,
       attribute units-per-em { xsd:string }?,
       attribute stemv { xsd:string }?,
       attribute stemh { xsd:string }?,
       attribute slope { xsd:string }?,
       attribute cap-height { xsd:string }?,
       attribute x-height { xsd:string }?,
       attribute accented-height { xsd:string }?,
       attribute ascent { xsd:string }?,
       attribute descent { xsd:string }?,
       attribute ideographic { xsd:string }?,
       attribute alphabetic { xsd:string }?,
       attribute mathematical { xsd:string }?,
       attribute hanging { xsd:string }?,
       attribute underline-position { xsd:string }?,
       attribute underline-thickness { xsd:string }?,
       attribute strikethrough-position { xsd:string }?,
       attribute strikethrough-thickness { xsd:string }?,
       attribute overline-position { xsd:string }?,
       attribute overline-thickness { xsd:string }?,
     (desc)*
     }
   metadata =
     element metadata {
       text
       }

   #---  End of SVG 1.2 PS RNC schema
```

### 7.1. Tools available for use with RNC

By no means an exhaustive list, but these were used while validating the RNC above. The tools must be used against an uncompressed version of the document.

* jing: Java-based RNG tool: https://relaxng.org/jclark/jing.html (https://relaxng.org/jclark/jing.html)

* pyjing: Python-based RNG tool: https://pypi.org/project/jingtrang/ (https://pypi.org/project/jingtrang/)

Example:

```
jing -c SVG-1.2-PS.rnc /path/to/file.svg
```

## 8. References

[SVG] W3C Recommendation, Scalable Vector Graphics (SVG) 1.1 (Second Edition), August 2011 https://www.w3.org/TR/2011/REC-SVG11-20110816/ (https://www.w3.org/TR/2011/REC-SVG11-20110816/)

[SVGT] W3C Recommendation, Scalable Vector Graphics (SVG) Tiny 1.2 Specification, December 2011 https://www.w3.org/TR/2008/REC-SVGTiny12-20081222/ (https://www.w3.org/TR/2008/REC-SVGTiny12-20081222/)

[RNC] RelaxNG Compact https://relaxng.org/compact-20021121.html (https://relaxng.org/compact-20021121.html)

## Authors' Addresses

Alex Brotman
Comcast, Inc
Email: alex_brotman@comcast.com

Trent Adams
Proofpoint, Inc
Email: tadams@proofpoint.com