Internet Architecture Board (IAB)
Request for Comments: 7996
Category: Informational
ISSN: 2070-1721

N. Brownlee
The University of Auckland

December 2016

SVG Drawings for RFCs: SVG 1.2 RFC

Abstract

This document specifies SVG 1.2 RFC -- an SVG profile for use in diagrams that may appear in RFCs -- and considers some of the issues concerning the creation and use of such diagrams.

Status of this Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Architecture Board (IAB) and represents information that the IAB has deemed valuable to provide for permanent record. It represents the consensus of the Internet Architecture Board (IAB). Documents approved for publication by the IAB are not a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7996[1].

Copyright Notice

[1] http://www.rfc-editor.org/info/rfc7996
[2] http://trustee.ietf.org/license-info

Table of Contents

1. Introduction

Over the last few years, the RFC Editor has worked with the Internet community to develop specifications for changes in the format of RFCs. An outline of the resulting specifications was published as [RFC6949] in May 2013. Since then, a Design Team has been working with the RFC Editor to flesh out those specifications. One aspect of the changes is to allow line drawings in RFCs; [RFC6949] says

> *Graphics may include ASCII art and a more complex form to be defined, such as SVG line art [SVG]. Color and grayscale will not be accepted. RFCs must correctly display in monochromatic black-and-white to allow for monochrome displays, black-and-white printing, and support for visual disabilities.*

SVG (Scalable Vector Graphics) has been developed by W3C, the World Wide Web Consortium; its current standard is SVG 1.1 Full [W3C.REC-SVG11-20110816]. This document defines SVG 1.2 RFC, an SVG profile (i.e., a subset of SVG) that is suitable for RFC line drawings.

Note that in RFCs, the text provides normative descriptions of protocols, systems, etc. Diagrams may be used to help explain concepts more clearly, but they provide supporting details and should not be considered to be complete specifications in themselves.

The details (particularly any vocabularies) described in this document are expected to change based on experience gained in implementing the new publication toolsets. Revised documents will be published capturing those changes as the toolsets are completed. Other implementers must not expect those changes to remain backwards-compatible with the details described in this document.

2. SVG 1.2 RFC: An SVG Profile for RFCs

As a starting point for SVG 1.2 RFC, the Design Team decided to use SVG Tiny 1.2 (also referred to as "SVG 1.2 Tiny") [W3C.REC-SVGTiny12-20081222]. SVG 1.2 Tiny is an SVG subset intended to be implemented on small, mobile devices such as cell phones and smartphones. That should allow RFCs to be rendered well on such devices, especially those that have small screens. However, RFCs are self-contained documents that do not change once they are published. The use of SVG drawings in RFCs is intended to allow authors to create drawings that are simple to produce and are easier to understand than our traditional "ASCII art" ones. In short, we are also trying to improve access to the content in RFCs, so SVG drawings need to be kept as simple as possible.

Appendix A (below) provides a complete RELAX NG Compact (RNC) schema [RNG-HOME] for SVG 1.2 RFC. It is derived from the SVG 1.2 schema, and is the formal definition of SVG 1.2 RFC. The remainder of this section gives a simplified -- i.e., easier to read and understand -- overview of SVG 1.2 RFC.

SVG can provide a complete User Interface, but within RFCs, all we need are simple diagrams that do not change once the RFC is published. Therefore, SVG 1.2 RFC does not allow anything from the following sections in SVG Tiny 1.2 [W3C.REC-SVGTiny12-20081222]:
12 Multimedia
13 Interactivity
15 Scripting
16 Animation
18 Metadata
19 Extensibility
Note that SVG Tiny 1.2 elements may have many properties or attributes that are needed to support aspects of the above sections. Those are not allowed in SVG 1.2 RFC.

We now consider these other sections in SVG Tiny 1.2 [W3C.REC-SVGTiny12-20081222]:

9 Basic Shapes	
10 Text	Everything in this section is allowed in SVG 1.2 RFC.
11 Painting: Filling, Stroking, Colors and Paint Servers	Anything relating to 'color' is not allowed in SVG 1.2 RFC; everything else is allowed. This

is a requirement documented in [RFC6949].

14 Linking

SVG Tiny 1.2 allows Internationalized Resource Identifiers (IRIs) in references. In SVG 1.2 RFC, such links must be ASCII only. That should not cause problems, since one can just use the URI form of any IRI. Authors should try to use links only to URIs that are

	long-term stable.
17 Fonts	SVG 1.2 RFC only allows 'serif', 'sans-serif', and 'monospace' generic font families from the WebFonts facility, described in Section 15 ("Fonts") of the W3C Cascading Style Sheets (CSS) 2.1 document [W3C.REC-CSS2-20110607]. In particular, the SVG 'font' element is not allowed.

2.1. Elements, Properties, and Attributes Allowed in SVG 1.2 RFC

This section discusses elements, properties, and attributes selected for SVG 1.2 RFC from [W3C.REC-SVGTiny12-20081222].

In the list below, elements and properties are listed on the left, and their allowed values are given in parentheses on the right.

<color> is the list of allowed colors, a black-and-white subset of the SVG color names.

<style> is a set of CSS attributes that are commonly used (by SVG drawing applications). They are not part of SVG Tiny 1.2 but are included here for compatibility. Note that

- There is no guarantee that any renderer will implement all the CSS attributes a drawing application may use.
- Authors will need to consider the compatibility of their drawings with rendering devices.

Elements:

svg	(version=1.2, baseProfile=tiny, width, viewBox, preserveAspectRatio, snapshotTime, height, id, role)
g	(label, class, id, role, fill, <style>, transform)
defs	(id, role, fill)
title	(id, role)
desc	(id, role)
a	(id, role, fill, transform)
use	(x, y, href, xlink:href, id, role, fill, transform)
rect	(x, y, width, height, rx, ry, stroke-miterlimit, id, role, fill, <style>, transform)
circle	(cx, cy, r, id, role, fill, <style>, transform)
ellipse	(cx, cy, rx, ry, id, role, fill, <style>, transform)
line	(x1, y1, x2, y2, id, role, fill, transform)
polyline	(points, id, role, fill, transform)
polygon	(points, id, role, fill, <style>, transform)
text	(x, y, rotate, space, id, role, fill, <style>, transform)
tspan	(x, y, id, role, fill)
textArea	(x, y, width, height, auto, id, role, fill, transform)
tbreak	(id, role)
solidColor	(id, role, fill)
linearGradient	(gradientUnits, x1, y1, x2, y2, id, role, fill)
radialGradient	(gradientUnits, cx, cy, r, id, role, fill)
stop	(id, role, fill)
path	(d, pathLength, stroke-miterlimit, id, role, fill, <style>, transform)

Properties: (most allow "inherit" as a value)

<style>	(font-family, font-weight, font-style, font-variant, direction, unicode-bidi, text-anchor, fill, fill-rule)
<color>	(black, white, #000000, #ffffff, #FFFFFF)
stroke	(<color>, none, currentColor)
stroke-width	
stroke-linecap	(butt, round, square)
stroke-linejoin	(miter, round, bevel)
stroke-miterlimit	
stroke-dasharray	
stroke-dashoffset	
stroke-opacity	
vector-effect	(non-scaling-stroke, none)
viewport-fill	(none, currentColor)
viewport-fill-opacity	
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)
visibility	(visible, hidden, collapse)

color-rendering	(auto, optimizeSpeed, optimizeQuality)
shape-rendering	(auto, optimizeSpeed, crispEdges, geometricPrecision)
text-rendering	(auto, optimizeSpeed, optimizeLegibility, geometricPrecision)
buffered-rendering	(auto, dynamic, static)
opacity	
solid-opacity	
solid-color	(currentColor, <color>)
color	(currentColor, <color>)
stop-color	(currentColor, <color>)
stop-opacity	
line-increment	(auto)
text-align	(start, end, center)
display-align	(auto, before, center, after)
font-size	
font-family	(serif, sans-serif, monospace)
font-weight	(normal, bold, bolder, lighter)
font-style	(normal, italic, oblique)
font-variant	(normal, small-caps)
direction	(ltr, rtl)
unicode-bidi	(normal, embed, bidi-override)
text-anchor	(start, middle, end)
fill	(none, <color>)
fill-rule	(nonzero, evenodd)
fill-opacity	

3. How to Create SVG Drawings

Many drawing packages can be used to create SVG drawings -- for example, Open Source packages Inkscape and Dia. Be aware that such packages may use SVG elements or attributes that are not allowed in SVG 1.2 RFC.

- For example, the 'marker' attribute is often used to place symbols such as arrowheads on lines, but 'marker' is not allowed in SVG 1.2 Tiny or SVG 1.2 RFC. In such cases, one has to draw the arrowhead in another, simpler way.
- SVG clip paths are used to define a shape; objects outside that shape become invisible. The 'clipPath' element is not allowed in SVG 1.2 Tiny or SVG 1.2 RFC.

Diagrams produced with these packages may contain elements, their attributes or properties, or values of attributes or properties that are not allowed in SVG 1.2 RFC. We will need to provide a tool to either (1) strip out anything that is not allowed in SVG 1.2 RFC or (2) replace disallowed values (e.g., replace 'Sans' with 'sans#serif' as an allowed value for 'font#family'). Experience with a simple test version of a tool for this has shown that such deletion and replacement can be effective for making SVG files from drawing packages conform to SVG 1.2 RFC, without visibly changing the diagrams they produce.

The tool described above can also be used by authors simply to check that their diagrams conform to SVG 1.2 RFC. To help with this, if visible changes do occur, the tool should produce a list of non#allowed keywords and the context in which they were found.

To include a diagram in an RFC, the xml2rfc (v3) tool will need to provide a way to include SVG drawings in Internet#Drafts, as described in Section 2.5 of [RFC7991].

4. Accessibility Considerations

One of the long-term goals for RFCs is to make them more accessible, e.g., to sight-impaired readers. For diagrams, it would be useful for authors to provide alternative forms of the diagram, so that voice-reading software could be used to "talk through" the diagram. Simply reading the SVG code for a complex diagram seems unlikely to work.

SVG 1.2 RFC allows SVG's 'title' and 'desc' elements. 'title' provides a brief text caption for an SVG object (much like a figure caption), and 'desc' provides a longer text description of what the object actually represents. As well, the SVG 'role' attribute can be used to indicate to a browser how an SVG object is to be interpreted. Good suggestions on how to use these elements are given in [SVG-ACCESS-TIPS].

ARIA is a W3C Recommendation for using SVG to create (as the name "ARIA" indicates) "Accessible Rich Internet Applications". A helpful introduction to ARIA is provided by [SVG-ARIA-PRIMER], while [SVG-USING-ARIA] gives examples of how to use ARIA to enhance SVG accessibility.

5. Examples of Diagrams Common in RFCs

Another way to create SVG drawings is to write programs to draw them. For example, using Python and its svgwrite module is a pleasant environment (for those who like writing code); this section presents a few examples of diagrams created in this way. Note that they are merely examples of typical diagrams from RFCs.

The SVG diagrams for this section, along with an HTML version of this document that includes the SVG diagrams, can be seen at [NB-SVG-1.2-RFC].

5.1. Packet Layout Diagrams

Example: Figure 3 from [RFC793].

5.2. Sequence Diagrams (1)

Example: Figure 6 from [ExpTrustedProxy].

5.3. Sequence Diagrams (2)

Example: Figure 3 from [RFC4321].

6. Security Considerations

This document does not introduce any security considerations on its own.

7. References

7.1. Normative References

[W3C.REC-SVGTiny12-20081222] Andersson, O., Berjon, R., Dahlstrom, E., Emmons, A., Ferraiolo, J., Grasso, A., Hardy, V., Hayman, S., Jackson, D., Lilley, C., McCormack, C., Neumann, A., Northway, C., Quint, A., Ramani, N., Schepers, D., and A. Shellshear, "Scalable Vector Graphics (SVG) Tiny 1.2 Specification", World Wide Web Consortium Recommendation REC-SVGTiny12-20081222, December 2008, <http://www.w3.org/TR/2008/REC-SVGTiny12-20081222>.

[W3C.REC-CSS2-20110607] Bos, B., Celik, T., Hickson, I., and H. Lie, "Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification", World Wide Web Consortium Recommendation REC-CSS2-20110607, June 2011, <http://www.w3.org/TR/2011/REC-CSS2-20110607>.

[RFC6949] Flanagan, H. and N. Brownlee, "RFC Series Format Requirements and Future Development", RFC 6949, DOI 10.17487/RFC6949, May 2013, <https://www.rfc-editor.org/info/rfc6949>.

7.2. Informative References

[W3C.REC-SVG11-20110816] Dahlstrom, E., Dengler, P., Grasso, A., Lilley, C., McCormack, C., Schepers, D., Watt, J., Ferraiolo, J., Fujisawa, J., and D. Jackson, "Scalable Vector Graphics (SVG) 1.1 (Second Edition)", World Wide Web Consortium Recommendation REC-SVG11-20110816, August 2011, <http://www.w3.org/TR/2011/REC-SVG11-20110816>.

[SVG-ACCESS-TIPS] Watson, L., "Tips for Creating Accessible SVG", May 2014, <http://www.sitepoint.com/tips-accessible-svg>.

[SVG-ARIA-PRIMER] Pappas, L., Schwerdtfeger, R., and M. Cooper, "WAI-ARIA 1.0 Primer", World Wide Web Consortium Working Draft, September 2010, <http://www.w3.org/TR/2010/WD-wai-aria-primer-20100916>.

[SVG-USING-ARIA] Watson, L., "Using ARIA to enhance SVG accessibility", The Paciello Group, December 2013, <https://www.paciellogroup.com/blog/2013/12/using-aria-enhance-svg-accessibility/>.

[RFC7991] Hoffman, P., "The "xml2rfc" Version 3 Vocabulary", RFC 7991, DOI 10.17487/RFC7991, December 2016, <http://www.rfc-editor.org/info/rfc7991>.

[NB-SVG-1.2-RFC] Brownlee, N., "Index of /materials/format/svg", <https://www.rfc-editor.org/materials/format/svg/>.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[ExpTrustedProxy] Loreto, S., Mattsson, J., Skog, R., Spaak, H., Bourg, G., Druta, D., and M. Hafeez, "Explicit Trusted Proxy in HTTP/2.0",

Work in Progress, draft-loreto-httpbis-trusted-proxy20-01, February 2014.

[RNG-HOME] Murata, M., "RELAX NG home page", February 2014, <http://www.relaxng.org/>.

[RFC4321] Sparks, R., "Problems Identified Associated with the Session Initiation Protocol's (SIP) Non-INVITE Transaction", RFC 4321, DOI 10.17487/RFC4321, January 2006, <https://www.rfc-editor.org/info/rfc4321>.

A. RELAX NG Compact (RNC) Schema for SVG 1.2 RFC

The following RNC schema can be used to check whether an SVG file conforms to SVG 1.2 RFC. For example, if this schema were contained in a file called SVG-1.2-RFC.rnc, the following command will test whether SVG file diagram.svg is a conformant SVG 1.2 RFC drawing.

```
jing -c SVG-1.2-RFC.rnc diagram.svg
```

The RNC schema included below is available on the RFC Editor website <https://www.rfc-editor.org/materials/format/SVG-1.2-RFC.rnc>. The website is considered definitive should there be any discrepancies.

```
#---  SVG 1.2 RFC RNC schema; Nevil Brownlee, Thu 26 Jan 2016 (NZST)

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

rfc-color = (  # SVG-1.2-RFC doesn't allow color or grayscale
  "black" | "white" | "#000000" | "#FFFFFF" | "#ffffff" | "inherit" )
start = svg
svg =
  element svg {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        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 { rfc-color }?,
        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 { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        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 {
```

IAB Members at the Time of Approval

The IAB members at the time this memo was approved were (in alphabetical order):

Jari Arkko
Ralph Droms
Ted Hardie
Joe Hildebrand
Russ Housley
Lee Howard
Erik Nordmark
Robert Sparks
Andrew Sullivan
Dave Thaler
Martin Thomson
Brian Trammell
Suzanne Woolf

Acknowledgements

Thanks to the RSE and the Design Team members for their helpful comments and suggestions for SVG 1.2 RFC. Thanks also to the wider IETF community for their input.

Author's Address

Nevil Brownlee
The University of Auckland
EMail: n.brownlee@auckland.ac.nz