



# Adobe<sup>®</sup> SVG Viewer 3.0 Extensions

*September 26, 2001*

**ADOBE SYSTEMS INCORPORATED**


**Corporate Headquarters**

345 Park Avenue

San Jose, CA 95110-2704

(408) 536-6000

<http://www.adobe.com>



Copyright © 2001 Adobe Systems Incorporated. All rights reserved.

NOTICE: All information contained herein is the property of Adobe Systems Incorporated. No part of this publication (whether in hardcopy or electronic form) may be reproduced or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Adobe Systems Incorporated.

Adobe, the Adobe logo, Acrobat, PostScript and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Apple, Macintosh, and QuickTime are trademarks of Apple Computer, Inc., registered in the United States and other countries. UNIX is a trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd. All other trademarks are the property of their respective owners.

***This publication and the information herein is furnished AS IS, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies, makes no warranty of any kind (express, implied, or statutory) with respect to this publication, and expressly disclaims any and all warranties of merchantability, fitness for particular purposes, and noninfringement of third party rights.***

# 1

# Adobe SVG Viewer 3.0 Extensions

## 1.1 About this document

This document describes the custom elements and attributes that are supported in the Adobe<sup>®</sup> SVG Viewer 3.0.

## 1.2 About Adobe SVG Viewer 3.0 Extensions

The Adobe SVG Viewer version 3.0 supports the following extensions to the SVG language:

- Audio
- Animation
- Transparency

## 1.3 Namespace

The following is the namespace URI for Adobe SVG Viewer 3.0 Extensions:

```
http://ns.adobe.com/AdobeSVGViewerExtensions/3.0/
```

## 1.4 DTD

The following is the DTD for the Adobe SVG Viewer 3.0 Extensions

```
<!-- AdobeSVGViewerExtensions.dtd (version 3.0) 20010915 -->

<!-- This DTD summarizes all supported extensions to SVG 1.0
supported by the Adobe SVG Viewer version 3.0.
The namespace URI is "http://ns.adobe.com/AdobeSVGViewerExtensions/3.0/"
This DTD is formulated under the expectation that a parent
DTD will set up the following entities:
    NSPREFIX-AV (Namespace prefix for this namespace, such as "a:")
    XMLNS-DECLARE-AV (xmlns[:prefix] attribute declaration)
    (and variable XLink entities)
and then include this DTD via reference. -->

<!-- Data type definitions . -->

<!ENTITY % URI "CDATA" >
<!ENTITY % Script "CDATA" >
<!ENTITY % number "CDATA" >
<!ENTITY % scriptImplementationValue "CDATA" >
    <!-- Valid values are 'Adobe', 'browser', 'Microsoft' or 'Netscape'.
    Default is 'browser' -->
```

```

<!-- =====
      Attributes on 'svg' element
      ===== -->
<!ENTITY % AdobeViewerSVGElementAttributes
"%NSPREFIX-AV;scriptImplementation %scriptImplementationValue; #IMPLIED
%NSPREFIX-AV;timeline (independent|inherit) #IMPLIED
%NSPREFIX-AV;pause (all|none|dependents) #IMPLIED" >

<!-- =====
      Attributes on 'script' element
      ===== -->
<!ENTITY % AdobeViewerScriptElementAttributes
"%NSPREFIX-AV;scriptImplementation %scriptImplementationValue; #IMPLIED" >

<!-- =====
      'audio', 'audio3d' and 'animateClock' elements
      ===== -->

<!-- Additional audio and animation elements -->
<!ENTITY % audioExt "" >
<!ENTITY % audioElement "%NSPREFIX-AV;audio" >
<!ENTITY % audio3dElement "%NSPREFIX-AV;audio3d" >

<!ELEMENT %audioElement; (desc|title|metadata %audioExt;)* >
<!ATTLIST %audioElement;
  %XMLNS-DECLARE-AV;
  id ID #IMPLIED
  xml:base %URI; #IMPLIED
  onbegin %Script; #IMPLIED
  onend %Script; #IMPLIED
  onrepeat %Script; #IMPLIED
  %XMLNS-DECLARE-XLINK;
  %xlinkShowAttribute; (other) #FIXED 'other'
  %xlinkActuateAttribute; (onLoad) #FIXED 'onLoad'
  %xlinkHrefAttribute; %URI; #IMPLIED
  type CDATA #IMPLIED
  begin CDATA #IMPLIED
  dur CDATA #IMPLIED
  end CDATA #IMPLIED
  restart (always | never | whenNotActive) 'always'
  repeatCount CDATA #IMPLIED
  repeatDur CDATA #IMPLIED
  fill (remove | freeze) 'remove'
  volume %number; #IMPLIED
  pan %number; #IMPLIED
  >

<!ELEMENT %audio3dElement; (desc|title|metadata %audioExt;)* >
<!ATTLIST %audio3dElement;
  %XMLNS-DECLARE-AV;
  id ID #IMPLIED
  xml:base %URI; #IMPLIED
  onbegin %Script; #IMPLIED
  onend %Script; #IMPLIED
  onrepeat %Script; #IMPLIED
  %XMLNS-DECLARE-XLINK;
  %xlinkShowAttribute; (other) #FIXED 'other'
  %xlinkActuateAttribute; (onLoad) #FIXED 'onLoad'
  %xlinkHrefAttribute; %URI; #IMPLIED
  type CDATA #IMPLIED
  begin CDATA #IMPLIED
  dur CDATA #IMPLIED

```

```

end CDATA #IMPLIED
restart (always | never | whenNotActive) 'always'
repeatCount CDATA #IMPLIED
repeatDur CDATA #IMPLIED
fill (remove | freeze) 'remove'
volume %number; #IMPLIED
x %number; #IMPLIED
y %number; #IMPLIED
z %number; #IMPLIED
vx %number; #IMPLIED
vy %number; #IMPLIED
vz %number; #IMPLIED
>

<!-- 'animateClock' controls the clock on a time container or an animation element.
Using the 'to' attribute set an absolute time value.
Using the 'by' attributes sets a relative time value.
Changing an animation element's clock means changing its begin time such that
the portion of the animation that is being displayed is offset into the animation
by the clock amount. For example, at t=20s let's say an animation is running that
started at 10s. Now, to="5s" would set the begin time to 15s; to="-5s" would set
begin to 25s (to start in the future); by="5s" would set the begin time to 5s (fast
forward by 5s); and by="-5s" would set begin to 15s (rewind by five seconds).
Note: no 'from' or 'values' -->
<!ENTITY % animateClockExt "" >
<!ENTITY % animateClockElement "%NSPREFIX-AV;animateClock" >

<!ELEMENT %animateClockElement; (desc|title|metadata %animateClockExt;)* >
<!ATTLIST %animateClockElement;
  %XMLNS-DECLARE-AV;
  id ID #IMPLIED
  xml:base %URI; #IMPLIED
  onbegin %Script; #IMPLIED
  onend %Script; #IMPLIED
  onrepeat %Script; #IMPLIED
  %XMLNS-DECLARE-XLINK;
  %xlinkShowAttribute; (other) #FIXED 'other'
  %xlinkActuateAttribute; (onLoad) #FIXED 'onLoad'
  %xlinkHrefAttribute; %URI; #IMPLIED
  begin CDATA #IMPLIED
  dur CDATA #IMPLIED
  end CDATA #IMPLIED
  restart (always | never | whenNotActive) 'always'
  repeatCount CDATA #IMPLIED
  repeatDur CDATA #IMPLIED
  fill (remove | freeze) 'remove'
  to CDATA #IMPLIED
  by CDATA #IMPLIED
  additive (replace | sum) 'replace'
  >

<!-- =====
                Adobe transparency properties
                ===== -->

<!-- Data types for Adobe transparency. -->

<!ENTITY % boolean_inherit "(true|false|inherit)" >
  <!-- All properties must allow for 'inherit' value. -->

<!ENTITY % iccname_inherit "CDATA" >
  <!-- Either icc-name(<name>) or 'inherit'. The <name> must
  match a name value from color profile dictionary as defined in an SVG
  <color-profile> element of @color-profile rule. -->

<!ENTITY % number-zero-to-one "CDATA" >

```

```

<!-- Adobe transparency properties.
Can be specified in CSS style sheets, SVG's 'style' attribute,
or as presentation attributes in the Adobe General namespace.
Here, the transparency properties packaged as presentation attributes
are presented as an entity declaration.
(For a definition of the term "presentation attributes",
refer to http://www.w3.org/TR/SVG/styling.html#UsingPresentationAttributes.) -->

<!-- Properties xxx-opacity-share are used to specify what fraction of
any *-opacity property from SVG maps to the Adobe Transparency Model (ATM)'s
concept of ATM-opacity vs. ATM-shape. Here are the formulas:
ATM-shape=svg-opacity^(1-adobe-opacity-share)
ATM-opacity=svg-opacity^adobe-opacity-share

(where a^x is a in the power of x, and a^0 is assumed to be one).
When you multiply ATM-shape and ATM-opacity you get SVG's opacity. -->

<!ENTITY % TransparencyPresentationAttributes
"%NSPREFIX-AV;adobe-blending-color-space
(DeviceGray|DeviceRGB|DeviceCMYK|CalGray|CalRGB|
sRGB|linearRGB|icc|inherit) #IMPLIED
%NSPREFIX-AV;adobe-blending-color-space-icc-name %iccname_inherit; #IMPLIED
%NSPREFIX-AV;adobe-blending-mode
(normal|multiply|screen|difference|darken|lighten|colorDodge|
colorBurn|exclusion|hardLight|overlay|softLight|luminosity|
hue|saturation|color|compatibleOverprint|inherit) #IMPLIED
%NSPREFIX-AV;adobe-isolated %boolean_inherit; #IMPLIED
%NSPREFIX-AV;adobe-knockout %boolean_inherit; #IMPLIED
%NSPREFIX-AV;adobe-knockout-text %boolean_inherit; #IMPLIED

%NSPREFIX-AV;adobe-opacity-share %number-zero-to-one; #IMPLIED
%NSPREFIX-AV;adobe-fill-opacity-share %number-zero-to-one; #IMPLIED
%NSPREFIX-AV;adobe-stroke-opacity-share %number-zero-to-one; #IMPLIED
%NSPREFIX-AV;adobe-stop-opacity-share %number-zero-to-one; #IMPLIED
" >

```