



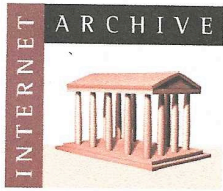
www.archive.org  
415 561 6767  
415 840-0391 e-fax

Internet Archive  
300 Funston Avenue  
San Francisco, CA 94118

---

## AFFIDAVIT OF DUNCAN HALL

1. I am a Records Request Processor at the Internet Archive, located in San Francisco, California. I make this declaration of my own personal knowledge.
2. The Internet Archive is a website that provides access to a digital library of Internet sites and other cultural artifacts in digital form. Like a paper library, we provide free access to researchers, historians, scholars, and the general public. The Internet Archive has partnered with and receives support from various institutions, including the Library of Congress.
3. The Internet Archive has created a service known as the Wayback Machine. The Wayback Machine makes it possible to browse more than 450 billion pages stored in the Internet Archive's web archive. Visitors to the Wayback Machine can search archives by URL (i.e., a website address). If archived records for a URL are available, the visitor will be presented with a display of available dates. The visitor may select one of those dates, and begin browsing an archived version of the Web. Links on archived files in the Wayback Machine point to other archived files (whether HTML pages or other file types), if any are found for the URL indicated by a given link. For instance, the Wayback Machine is designed such that when a visitor clicks on a hyperlink on an archived page that points to another URL, the visitor will be served the archived file found for the hyperlink's URL with the closest available date to the initial file containing the hyperlink.
4. The archived data made viewable and browseable by the Wayback Machine is obtained by use of web archiving software that automatically stores copies of files available via the Internet, each file preserved as it existed at a particular point in time.
5. The Internet Archive assigns a URL on its site to the archived files in the format `http://web.archive.org/web/[Year in yyyy][Month in mm][Day in dd][Time code in hh:mm:ss]/[Archived URL]` aka an "extended URL". Thus, the extended URL `http://web.archive.org/web/19970126045828/http://www.archive.org/` would be the URL for the record of the Internet Archive home page HTML file (`http://www.archive.org/`) archived on January 26, 1997 at 4:58 a.m. and 28 seconds (1997/01/26 at 04:58:28). A web browser may be set such that a printout from it will display the URL of a web page in the printout's footer. The date indicated by an extended URL applies to a preserved instance of a file for a given URL, but not necessarily to any other files linked therein. Thus, in the case of a page constituted by a primary HTML file and other separate files (e.g., files with images, audio, multimedia, design elements, or other embedded content) linked within that primary HTML file, the primary HTML file and the other files will each have their own respective extended URLs and may not have been archived on the same dates.
6. Attached hereto as Exhibit A are true and accurate copies of browser printouts of the Internet Archive's records of the archived files for the URLs and the dates



specified in the footer of the printout or an attached coversheet (in the case of records for which a browser does not provide a ready option to print a URL in the footer, e.g., in the case of a PDF file).

7. I declare under penalty of perjury that the foregoing is true and correct.

DATE: Nov 11, 2020

Duncan Hall  
Duncan Hall

# EXHIBIT A

http://msdn.microsoft.com/workshop/imedia/windowsmedia/ad.asp Go SEP OCT NOV 13 1998 1999 2000 About this capture

30 captures  
13 Oct 1999 – 26 Oct 2008

msdn online  
**Web Workshop**

All Products | Support | Search | microsoft.com Home

Microsoft

HOME | VOICES | LIBRARIES | COMMUNITY | DOWNLOADS | SITE GUIDE | SEARCH MSDN

 [show toc](#)  [index](#)

[Web Workshop](#) | [Streaming & Interactive Media](#) | [Windows Media Technologies](#)

# Ad Insertion

December 14, 1998

- [Introduction](#)
- [Alphabet Soup Decoded](#)
- [Example One: Simple Ad Insertion](#)
- [Example Two: Simple ASX file with enhanced elements](#)
- [Example Three: Dynamic ASX file via ASP](#)
- [Summary](#)

## Introduction

Streaming media is an example of compelling content on the Internet. **Adding advertisements using streaming media** is a great way to communicate commercial messages. In this article, we'll learn how to insert ads into a play list, and expand the idea to utilize dynamic ad personalization by using Active Server Page technology (ASP).

## Alphabet Soup Decoded

There are many file formats and acronyms you must become familiar with for **Ad Insertion**:

- Microsoft's native file format for streaming media is known as Advanced Streaming Format ([ASF](#)). This low-overhead storage and transmission file format encapsulates multimedia data types (images, audio, and video) as well as embedded text (URLs, for example), and allows for the synchronization of these objects within a stream.  
Individual ASF clips can be combined into *play lists* by creating [ASX files](#). These files use industry-standard Extensible Markup Language (XML) syntax. The ASX file is linked from the ASF file and basically describes each clip.
- Active Server Pages (ASP), are dynamic Web pages used in conjunction with Microsoft IIS (Internet Information Server). ASP is an environment in which you can combine HTML, scripts, and reusable ActiveX server components to create dynamic and powerful Web-based business solutions. ASP's enable server-side scripting for IIS with native support for both VBScript and Jscript.

Now, see the examples below. You are well on your way to adding ads.

[Back to top](#)

## Example One: Simple Ad Insertion

In our first example, we will insert an ad in front of a single video clip. All visitors will see the same ad, and the same video clip.

```
<ASX Version="3">

  <ENTRY ClientSkip="no">
    <REF HREF="mms://127.0.0.1/AdInsertion/HollandAmerica200.asf"/>
  </ENTRY>

  <ENTRY>
    <REF HREF="mms://127.0.0.1/AdInsertion/NukeTest200.asf"/>
  </ENTRY>

</ASX>
```

The ASX elements are simple. This code defines the file as an ASX file, and tells Windows Media Player what version you're using.

```
<ASX Version="3">
```

Then we define two entries. Note that the first entry (the advertisement) has a special descriptor, `ClientSkip="no"`, that prevents the site visitor from skipping the ad. In order to enforce this behavior, the player disables three controls: [Next Clip](#), [Preview Mode](#), and the [slider bar](#).

After viewing the ad, the site visitor will immediately see the second entry with normal player behavior restored. How this works is the Windows Media Player starts buffering the second clip while the advertisement is still playing, which eliminates the traditional "black hole" between clips. This behavior is unique to Microsoft's media format, and is a feature customers appreciate.

Finally, the `</ASX>` tag marks the end of this file.

[↩ Back to top](#)

## Example Two: Simple ASX File with Enhanced Elements

Example One showed the simplest advertisement scenario. Example Two will expand on this by adding [titles](#), [abstracts](#), and a ["Buy Now"](#) button. The Buy Now button can be used either to allow users to purchase content, or simply to send them to the advertiser's Web site. Examine the following code:

```
<ASX version="3">

<TITLE>Demo ASX File</Title>

<ENTRY ClientSkip="no">
  <REF HREF="mms://Raratonga/AdInsertion/HollandAmerica200.asf"/>
  <banner HREF="http://Raratonga/AdInsertion/images/MoreInfo.gif">
    <moreinfo HREF="http://www.hollandamerica.com"
  </BANNER>
  <TITLE>Holland America Travel</TITLE>
  <COPYRIGHT> 1998, Holland America </COPYRIGHT>
</ENTRY>

<ENTRY>
  <TITLE>Secret US Nuke Tests</TITLE>
```

```

<COPYRIGHT> Copyright MSNBC 6/11/98 </COPYRIGHT>
<REF HREF="mms://Raratonga/AdInsertion/NukeTest200.asf"/>
<ABSTRACT>
::Recently declassified footage shows that the United States conducted secret nuclear
weapons testing in the upper stratosphere just before the nuclear
weapons ban went into
effect. Tests were not always successful, with at least one rocket
exploding on the launch
pad.
<ABSTRACT>
<COPYRIGHT> Copyright 1998, MSNBC -- All Rights Reserved </COPYRIGHT>
lt;ENTRY>

</ASX>

```

Let's look at each of the new elements. The <TITLE> element describes a show title, which appears in the standalone player (see Figure 1 below). All of the elements described can also be used in conjunction with the embedded player, and rendered via JavaScript.


The <BANNER> tag points to a GIF for the button image which points to a URL. Notice how that target is used to point to a frame name, just like you would in a normal HTML page. And, like a normal HTML page, in-line JavaScript can be used in place of a URL.

The <ABSTRACT> tag describes Tool Tip text that appears when the user's mouse hovers over the clip title. <COPYRIGHT>, <TITLE>, and <AUTHOR> tags describe various display elements in the player also.

[↩ Back to top](#)

## Example Three: Dynamic ASX Files via ASP

The most compelling advertisements are personalized to user preferences. Users interested in travel on cruise ships won't see backpacking ads, and vice-versa. This avoids annoying viewers, and at the same time commands premium rates from advertisers because they know only interested viewers will see their ad.

[Microsoft AdServer](#) , a component of Microsoft SiteServer, is one way to schedule these ads, but your **personalization can be as simple or as robust as you find necessary.**

Example Three will use a simple radio button selection that chooses one of two ads. This, in turn, will drive dynamic content and will focus on how ASP gets generated.

### Example

There are three required steps to creating a dynamic play list:

1. **Ask the user for personalized information.** We use a simple form with two choices in our example below.
2. **Process the user preferences to create a personalized play list.** We use ASP to create the list and the Windows Media Player as an embedded object. The first step calls the page up containing the player reference. That page then calls a second ASP page up that returns the ASX file.
3. **Play the user's content.** Presto!

The user preference form (entitled *preferences.htm*) is shown below. The form method is set to **post** in order to make the URL cleaner for *PlayerPage.asp*.

&lt;HTML&gt;

&lt;! Preferences.htm&gt;

&lt;BODY&gt;

Please choose one of the following:&lt;BR&gt;

&lt;FORM METHOD=POST ACTION=PlayerPage.asp&gt;

&lt;INPUT TYPE=RADIO NAME=AdPref VALUE=0&gt; I like travel

&lt;BR&gt;

&lt;INPUT TYPE=RADIO NAME=AdPref VALUE=1&gt; I like backpacking

&lt;P&gt;

&lt;INPUT TYPE=SUBMIT&gt;

&lt;/FORM&gt;

&lt;/BODY&gt;

&lt;/HTML&gt;

*PlayerPage.asp* embeds the player, then passes user choices on to *BuildAsx.asp*.

&lt;%

Option Explicit

```
` This function will process as many arguments as are passed in
` the Request.form object. In this example the function is overkill,
` but is listed here as cookbook code for your use
```

Function UrlArgs()

Dim item, sTemp

For each item in Request.Form

` Keep concatenating data

sTemp = item &amp; "=" &amp; Request.Form (item) &amp; "&amp;"

Next

If Len(sTemp) &gt; 0 Then

` Trim the trailing &amp;, if there is data in sTemp

sTemp = Left (sTemp, Len (sTemp) - 1)

End If

UrlArgs = sTemp

End Function

%&gt;

&lt;html&gt;

&lt;body&gt;

&lt;OBJECT

ID=MediaPlayer1

Name="MediaPlayer1"

CLASSID="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"

```
codebase="http://activex.microsoft.com/activex/controls/mpplayer/en/nsmp
2inf.cab#Version=5, 1, 52, 701"
```

HEIGHT=412

WIDTH=326

&gt;

&lt;PARAM NAME="FILENAME"

VALUE="BuildAsx.asp?&lt;%=UrlArgs() %&gt;"&gt;

&lt;PARAM NAME="AutoStart"        VALUE="-1"&gt;

&lt;PARAM NAME="ShowControls"    VALUE="-1"&gt;

&lt;PARAM NAME="ShowTracker"    VALUE="-1"&gt;

&lt;PARAM NAME="Autosize"        VALUE="-1"&gt;

&lt;PARAM NAME="ShowStatusBar"  VALUE="-1"&gt;

&lt;PARAM NAME="ShowDisplay"    VALUE="-1"&gt;

&lt;EMBED TYPE="application/x-mplayer2"

pluginspage="http://www.microsoft.com/windows/mediaplayer/download/default.asp"

```

        id=MediaPlayer1
        Name="MediaPlayer1"
        DisplaySize="4" //Fit To Size
        AutoSize="-1"
        BgColor="darkblue"
        ShowControls="-1"
        ShowTracker="-1"
        ShowDisplay="-1"
        ShowStatusBar="-1"
        VideoBorder3D="-1"
        WIDTH=320
        Height=313
        SRC="BuildAsx.asp?<%=UrlArgs() %>"
        autostart="-1"
        DESIGNTIMESP="5311"
    >
</EMBED>
</OBJECT>

</body>
</html>

```

[⬆ Back to top](#)

The real magic is performed in *BuildAsx.asp*. This page does not send the usual HTTP headers to the player. That is why the page works as an ASX file.

```

<%@ LANGUAGE="VBScript" %>
<%
-----
` Set the BASE_PATH Const below to point to your
` Windows Media Services server
-----
Option Explicit
Const BASE_PATH = "mms://media.MyDomain.com/AdInsertion/"

`Note that Request.ContentType is the "secret handshake"
`that allows us to create dynamic
`ASX files that do not contain HTTP header data.
Response.ContentType = "video/s-ms-asf" ` Set MIME type
Response.Expires = 0 `Force content to always refresh

Response.Write AsxHeader() `Create the opening ASX tag
Response.Write AdEntry() `Create an ad
Response.Write AsxEntry() `Create an XML entry for content
Response.Write AsxFooter() `Write the ASX footer
`And we're done!
-----
` Function AsxHeader
` This function returns an opening <ASX> tag
-----
Function AsxHeader ()
    Dim sTemp
    sTemp = "<ASX Version=""3"">" & vbNewLine & " " & vbNewLine
    sTemp = sTemp & "<TITLE> Demo Dynamic ASX File From Active " _
        & "Server Pages </Title>" & vbNewLine & " " & vbNewLine
    ASXHeader = sTemp `return the string
End Function
-----
` Function AdEntry
` This function returns an ad based on user request
-----
Function AdEntry()
    Dim sTemp
    sTemp = "<ENTRY>" & vbNewline

```



```

If cInt (0 & Request ("AdPref")) = 1 Then
    sTemp = sTemp & "          <REF HREF="" & BASE_PATH_
        & "BackpackingAd.asf"" />" & vbNewLine
Else
    sTemp = sTemp & "          <REF HREF="" & BASE_PATH_
        & "TravelAd.asf"" />" & vbNewLine
End If
sTemp = sTemp & "</entry>" & vbNewLine & vbNewLine
AdEntry = sTemp
End Function
\-----
\ Function AsxEntry
\ This function returns an XML entry for a specific story.
\ In real life this function will probably do a database lookup
\-----
Function AsxEntry ()
    Dim sTemp, sAbstract
    sTemp = "<ENTRY>" & vbNewLine
    sTemp = sTemp & "          <TITLE> Secret US Nuke Tests </TITLE>" & vbNewLine
    sTemp = sTemp & "          <REF HREF="" & BASE_PATH & "Nuketest200.asf""
/> & vbNewLine
    sTemp = sTemp & "          <ABSTRACT>" & vbNewLine
    sAbstract = "Recently declassified footage shows that the United
States conducted secret nuclear weapons testing in the upper
stratosphere just before the nuclear weapons ban went into effect.
Tests were not always successful, with at least one rocket exploding on
the launch pad."
    sTemp = sTemp & sAbstract & vbNewLine
    sTemp = sTemp & "          </ABSTRACT>" & vbNewLine
    sTemp = sTemp & "          <COPYRIGHT>Copyright 1998, MSNBC -- All Rights
Reserved </COPYRIGHT>" & vbNewLine
    sTemp = sTemp & "</ENTRY>" & vbNewLine & vbNewLine
    AsxEntry = sTemp
End Function
\-----
\ Function AsxFooter
\ This function returns a closing </ASX> tag
\-----
Function AsxFooter ()
    AsxFooter = "</ASX>" & vbNewLine & vbNewLine
End Function

%>

```

## Summary

The examples above are great cookbook code samples to get you started. Either use a sample "as is," or build on them to create really dynamic Web pages.

Additional examples can be found at [Windows Media Technologies Technical Showcase](#). Click on *Code Samples* and use the "view source" feature to see how we did it. Also be certain that you download the [client SDK](#). Here, there are even more examples and ideas shown, including examples that work in both Netscape's and Microsoft's browsers.

[Back to top](#)

*Did you find this material useful? Gripes? Compliments? Suggestions for other articles? [Write us!](#)*

© [1999 Microsoft Corporation. All rights reserved. Terms of use.](#)

http://msdn.microsoft.com/workshop/imedia/windowsmedia/crcontent/ Go APR MAY AUG 08 1998 1999 2000 About this capture

80 captures  
8 May 1999 - 31 Oct 2019

msdn online  
**Web Workshop**

All Products | Support | Search | microsoft.com Home

Microsoft

HOME | VOICES | LIBRARIES | COMMUNITY | DOWNLOADS | SITE GUIDE | SEARCH MSDN

 [show toc](#)  [index](#)

[Web Workshop](#) | [Streaming & Interactive Media](#) | [Windows Media Technologies](#)

# All about ASX Files

Updated February 4, 1999

- [Introduction to ASX Files](#)
- [Creating a Simple ASX](#)
- [File or Clip Properties](#)
- [Custom Graphics in the Windows Media Player](#)
- [Ad Insertion](#)
- [Playlists](#)
- [Server or Protocol Rollover](#)
- [Putting It All Together](#)
- [ASX Tags List](#)
- [ASX Files for Multicast](#)

## Introduction to ASX Files

ASX files are simply text files that act as links from Web pages to ASF files on Windows Media Services or HTTP servers. They transfer control of the data from the HTTP browser to the Windows Media Player so that the data can stream.

When a user hits a link to an ASX file, the user's browser downloads the entire ASX file to the user's cache directory (don't worry, the files are tiny -- about 1K in size). Then the user's computer looks in its file associations table and sees that when it hits an ASX file it should launch the Windows Media Player. The Windows Media Player launches, looks in the ASX for instructions on where to get the ASF file, and then starts the stream playing.

If you have an ASF file that you want users to be able to access, creating an ASX file is simple. This ASX file can be used to access on-demand ASF files, ASF files that contain live audio or video streams generated from a Windows Media Encoder, or Windows Media Services Station (.NSC) files.

[Back to top](#)

## Creating a Simple ASX File

To get started and create your first ASX, just go into your favorite text editor, such as Notepad, and type the following:

```
<ASX version="3">
  <Entry>
    <ref HREF="path" />
  </Entry>
</ASX>
```

Just substitute the path of the ASF file you have, where *path* is one of the following, depending on where you are streaming the content from:

### Source of ASF

ASF on Windows Media Services

Windows Media station

Publishing point on Windows Media Services

ASF on HTTP server

ASF on network server

ASF on local hard drive

### Syntax

*mms://servername/path/asfname.asf*

*http://servername/stations/kxyz.nsc*

*mms://servername/publishingpoint/*

*http://servername/path/asfname.asf*

*file://\servername\path\asfname.asf*

*file://c:\path\asfname.asf*

Once you've entered this into Notepad, save the file as *filename.asx*. Check to make sure that the ASX is working by double-clicking on it in Windows Explorer. It should bring up the Windows Media Player and start the content streaming. Once you've tested that it's working, just save it to your standard HTTP server along with your HTML pages, and link to it by means of a normal `<a href>` tag, or embed it in an HTML page using the [Windows Media <OBJECT> tag](#).

[⬆ Back to top](#)

## File or Clip Properties

When you play content served by Windows Media Technologies, you can right-click on the viewing area or video rendering area, and a drop-down menu will come up. Select Properties to view file information about the show or clip that is playing. The content creator can insert file properties into the ASF file when they create it, or they can set the properties in the ASX file that is used to link to their ASF. They can also set properties for an entire playlist that contains individual ASF files, each with its own properties.

Here is an example of an ASX file that inserts file properties (in the main section of the ASX) and clip properties (in individual `<ENTRY>` tags). In this case, the properties set are the [Abstract](#), [Title](#), [Author](#), [Copyright](#), and [MoreInfo](#), which allows the user to click on either a banner showing on the player and have it send their browser to that URL, or allows the user to click on a MoreInfo link on the clip or file properties for more information.

```
<ASX version = "3.0">
  <ABSTRACT>This text will show up as a Tooltip and in the Properties
  dialog box for the file</ABSTRACT>
  <TITLE>Title for this file</TITLE>
  <AUTHOR>The Name of the Author for this File</AUTHOR>
  <COPYRIGHT>1998 by Your Company</COPYRIGHT>
  <MoreInfo href = "http://www.microsoft.com/netshow" />
  <Entry>
    <Ref href = "mms://netshow.microsoft.com/sbnasfs/control.asf" />
    <MoreInfo href = "http://www.microsoft.com/netshow" Banner =
    "http://servername/path/banner1.gif" Style = "IMAGE">
    <Abstract>This is the description for this clip.</Abstract>
    </MoreInfo>
  </Entry>
</ASX>
```

```

<Title>Markers Discussion</Title>
<Copyright>1998 Microsoft Corporation</Copyright>
<Logo href = "http://servername/path/banner2.gif" Style = "ICON" />
<MoreInfo href = "http://www.microsoft.com/netshow"></MoreInfo>
<Ref href = "mms://netshow.microsoft.com/sbnasfs/marker.asf" />
<Ref href = "http://netshow.microsoft.com/sbnasfs/marker.asf" />
</Entry>
</ASX>

```

[↩ Back to top](#)

## Custom Graphics in the Windows Media Player

The Windows Media Player and Windows Media Services together allow you to add graphics elements to the Windows Media Player as the stream plays. There are four basic types of graphics you can add:

Type	Description
Banner	Allows you to place a banner (82 pixels x 30 pixels) image on the player underneath the content for Windows Media Services. This image is clickable. Done using the <a href="#">&lt;MoreInfo&gt;</a> tag.
Icon	Allows you to place a small (16 pixels x 16 pixels) image next to the Title of the Show, or next to the Show status on the player. This graphic replaces the icon displayed on the Display panel next to the Show or Clip title (replaces graphic next to Show title when placed in the main area of the ASX, and replaces the graphic next to the clip title when placed in an <a href="#">&lt;ENTRY&gt;</a> tag). Accomplished through the use of the <a href="#">&lt;LOGO&gt;</a> tag.
Image	Allows you to place a graphic (82 pixels x 30 pixels) on the player underneath the content for Windows Media Services. This graphic stays up for the duration of the show if placed in the main section of the ASX, or for the duration of the clip if placed within the <a href="#">&lt;ENTRY&gt;</a> tag. Graphic is displayed centered horizontally, six pixels (the size of the border) above the edge of the Video area. The "IMAGE" style graphic is not displayed if a Banner graphic is defined in a MoreInfo element. Accomplished through the use of the <a href="#">&lt;LOGO&gt;</a> tag.
Watermark	Allows you to place a banner or graphic (82 pixels x 30 pixels) image on the player while the content is buffering. Note that the watermark does not appear on top of the content, but on the ASF before or after it plays. The "MARK" style is only valid if the parent element of the Logo element is ASX. The "MARK" style graphic is displayed in the lower right corner of the Video area (allowing for a six-pixel border) while the player is connecting to a server and opening a piece of content. Accomplished through the use of the <a href="#">&lt;LOGO&gt;</a> tag.

The Icon, Image, and Mark graphics are added through the [<LOGO>](#) tag in the ASX file. Here's an example for each:

```

<Logo href = "http://server/logos/image1.gif" Style = "ICON" />
<Logo href = "http://server/logos/image1.gif" Style = "IMAGE" />
<Logo href = "http://server/logos/image1.gif" Style = "MARK" />

```

If you want to make it so a viewer can click on a banner ad and then have their browser send them to a specific URL, add the [<MoreInfo>](#) tag and specify a location to get the banner from:

```

<ASX version = "3.0">
<!-- This is a comment. The MoreInfo tag below makes the Show title
in the Display panel clickable. If the MoreInfo tag is under an Entry
element, the title of the clip can be clicked on.
When a user places the mouse over the title, the URL defined in the href
attribute is displayed in the Status bar. When a user clicks the title,
the browser starts and navigates to the page defined in the URL. When

```

the Banner attribute is used, the user can click the graphic in the Video area to open the URL.

End comment. -->

```
<MoreInfo href = "http://www.server.com/info" Banner
"http://www.server.com/logos/banner.jpg">
</MoreInfo>
<Entry>
  <Ref href = "mms://nserver/content/title1.asf" />
</Entry>
</ASX>
```

You can also use [Microsoft Site Server](#), with its Personalization and Ad Server functionality, to customize the advertisements the user receives. Simply point the URL for the advertisements to the URL of the Ad Server containing the graphics.

[Back to top](#)

## Ad Insertion

You can send script events down a live stream from the Windows Media Encoder that tell the ASX file to insert a specific clip into the stream. For example, at a ball game where the feed is being broadcast live over the Internet, when a team calls a time out, the person running the Windows Media Encoder could send an event down the stream called Time-Out, that when received by the client is acted upon in the ASX file:

```
<Event Name = "Time-Out" WhenDone = "RESUME" Refresh = "yes">
  <EntryRef href = "mms://nserver/content/advert.asf"
  ClientSkip = "no" />
</Event>
```

When the event named Time-Out is received by the client, the ASX file instructs it to insert an advertising clip called advert.asf into the stream, and when it's done playing, to rejoin the previous stream.

This can be especially powerful when you consider that you can use Active Server Pages (ASP) as links to the content. ASPs could automatically serve up ASF content that is personalized to the user's region or profile.

See "[Ad Insertion](#)" for more information.

[Back to top](#)

## Playlists

Playlists are one of the most powerful features of Windows Media Technologies. Playlists allow you to schedule content to play in succession, or to insert advertising or special interest clips into a stream after a specific period of time or at a specific point. One of the nice parts about playlists is that instead of playing an ASF, stopping, starting the next ASF, and then waiting for it to finish buffering, Microsoft Windows Media Services and the Windows Media Player work together to play the clips one after the other with minimal buffering time or interruption in between.

The most simple playlists are created by adding multiple Entry tags to the ASX file. Within each Entry tag you can have multiple Ref tags to specify alternative locations or protocols to get the content from ([protocol or server rollover](#)). Here's an example:

```
<ASX version = "3.0">
<Title>Title</Title>
<Entry><Ref href = "mms://nserver/content/title1.asf" /></Entry>
<Entry><Ref href = "mms://nserver/content/title2.asf" /></Entry>
<Entry><Ref href = "mms://nserver/content/title3.asf" /></Entry>
```

```
<Entry><Ref href = "mms://nsserver/content/title4.asf" /></Entry>
</ASX>
```

[⬆ Back to top](#)

## Server or Protocol Rollover

If you want to set it up so that it automatically rolls over to a different server, protocol, or path if a failure occurs, just add <REF HREF> statements:

```
<ASX version="3">
  <Entry>
    <ref HREF="mms://netshow.microsoft.com/sbnasfs/keister.asf"/>
    <ref HREF="mms://server/path/asfname.asf"/>
    <ref HREF="http://server2/path/asfname.asf"/>
  </Entry>
</ASX>
```

In this case, it tries keister.asf on the netshow.microsoft.com server, and if that fails it tries asfname.asf on *server* using the mms protocol, and if that fails, it rolls over to asfname.asf on *server2* using the http protocol. Keep in mind that Microsoft Windows Media Services can stream via UDP or HTTP, and even if no rollover servers are specified in the ASX, the player will automatically try protocols that are enabled on the client before it will look in the ASX for a different server to try.

[⬆ Back to top](#)

## Putting It All Together

Here's an example of an ASX that includes a variety of functionalities:

- File and clip properties set through [Abstract](#), [Title](#), [Author](#), [Copyright](#), and [MoreInfo](#)
- Relative paths for ASX files and graphics set using [RefBase](#)
- Link from banner ad to a Web page set using <MoreInfo href = "http://www.microsoft.com/netshow" Banner = "http://servername/path/banner1.gif" Style = "IMAGE">
- Banners, watermark, and icon to set [custom graphics in the player](#)
- Play list through placing multiple <ENTRY> tags in succession; first ASF plays for only 30 seconds by setting [<DURATION>](#)
- Server and protocol rollover by placing multiple [<REF>](#) tags in succession within one [<ENTRY>](#) tag
- Setting the clip so the user can't fast-forward through it by setting ClientSkip="no" in the [<ENTRY>](#) tag

```
<ASX version = "3.0" BannerBar = "FIXED">
  <ABSTRACT>This text will show up as a ToolTip for the file</ABSTRACT>
  <TITLE>Title for this file</TITLE>
  <AUTHOR>The Name of the Author for this File</AUTHOR>
  <COPYRIGHT>1998 by Your Company</COPYRIGHT>
  <Logo href = "http://servername/path/icon.jpg" style = "ICON" />
  <Logo href = "http://servername/path/image.jpg" style = "IMAGE" />
  <Logo href = "http://servername/path/mark.jpg" style = "MARK" />
  <RefBase HREF="http://servername/path/" />
  <MoreInfo href = "http://www.microsoft.com/netshow" />
  <Entry ClientSkip = "no">
    <Ref href = "mms://servername/path/asf1.asf" />
    <Ref href = "mms://servername2/path/asf1.asf" />
    <Ref href = "http://servername/path/asf1.asf" />
    <Duration value = "00:00:30" />
    <MoreInfo href = "http://www.microsoft.com/netshow" Banner =
      "http://servername/path/banner1.gif" Style = "IMAGE">
  <Abstract>This is a mock Windows Media endorsement.</Abstract>
```



```

    </MoreInfo>
  </Entry>
<Entry>
  <Title>Markers Discussion</Title>
  <Copyright>1998 Microsoft Corporation</Copyright>
  <Logo href = "http://servername/path/banner2.gif" Style = "ICON" />
  <MoreInfo href = "http://www.microsoft.com/netshow"></MoreInfo>
  <Ref href = "mms://netshow.microsoft.com/sbnasfs/marker.asf" />
  <Ref href = "http://netshow.microsoft.com/sbnasfs/marker.asf" />
</Entry>
</ASX>

```

[⏮ Back to top](#)

## ASX Tags List

Here's a recap of the key tags in the ASX files, and their descriptions and syntax.

### Abstract tag

#### Example:

```

<ASX version = "3.0">
  <abstract>
    This text will show up as a ToolTip
  </abstract>
  <Title>Hover over this title for a ToolTip.</title>
</ASX>

```

**Description:** Contains text that represents a description of the associated ASX or Entry element. This text is displayed by the player in both the Display panel and the Properties dialog box. If the Abstract element is used in an ASX element, the text is displayed as a ToolTip when the mouse is placed over the title of the show in the Show bar. The title displayed in the Show bar comes from the Title element in an ASX element. The text in the Abstract element also is displayed under Description on the Show tab of the Properties dialog box. When you use the Abstract element inside an Entry element in the .asx file, the text is displayed as a ToolTip when the mouse is placed over the clip title in the Display panel, and under Description on the Clip tab of the Properties dialog box.

### ASX tag

#### Example:

```

<asx version = "3.0" previewmode = "yes">
  <EntryRef href = "http://server/content/program1.asf" />
</asx>

```

**Description:** The ASX element defines a file as an .asx file. The first four characters of an .asx file must be <asx. Other elements, such as Title and Author, defined within the scope of the ASX element (where ASX is the parent element) are associated with the Show information displayed by Windows Media Player.

### Author tag

#### Example:

```

<author>
  Author Name
</author>

```



**Description:** The Author element is a text string representing the name of the author of the .asx file. You can use the Author element in the ASX and Entry elements. The text contained in the Author element is displayed in the Display panel and Properties dialog box of the player. If the Author element is used in the ASX element, the text is displayed on the Show tab of the Properties dialog box. You also can use the Author element in the Entry element, in which case the text is displayed under Clip Author in the Display area and on the Clip tab of the Properties dialog box.

## Base tag

### Example:

```
<Base HREF="http://www.servername.com/" />
```

**Description:** The Base element defines a Uniform Resource Locator (URL) string appended to (the front of) URL-flip URLs sent to the client. The Base element is similar to a home directory for relative links. The Base element affects only URLs sent to the client using script commands. URL-flipping is a scripting mechanism you can use to cause the player to flip to a new URL when a script command is received.

## Copyright tag

### Example:

```
<Copyright>Copyright 1998, CompanyName®</Copyright>
```

**Description:** The Copyright element is a text string that specifies the copyright information for an ASX or Entry element. The copyright information is displayed in the Details area of the player and the Properties dialog box. When used in the ASX element the copyright information is displayed only on the Show tab of the Properties dialog box. If you use the Copyright element in an Entry element, the text is displayed in the Clip bar of the Display area and on the Clip tab of the Properties dialog box.

## Duration tag

### Example:

```
<Duration value = "00:00:30" />  
<Duration value = "1:01.5" /> This is the same as 61.5 seconds.
```

**Description:** The Value attribute for the Duration element defines the length of time a stream is to be rendered by the client. It is possible to set the value of the Value attribute to a length of time that exceeds the end of the content stream, in which case the stream terminates normally.

## EndMarker tag

### Example:

```
<EndMarker number = "17" />  
<EndMarker name = "MarkerH" />
```

**Description:** The EndMarker element defines the named or numeric marker index where the player is to stop rendering the stream defined in the associated Entry or Ref element. An associated element is the Parent element of an element.

## Entry tag

### Example:

<https://web.archive.org/web/19990508200218/http://msdn.microsoft.com/workshop/imedia/windowsmedia/crcontent/asx.asp>

```

<ASX Version = "3.0">
<Title>The Something Catchy Station on Our Site</Title>
<MoreInfo href = "http://www.server.com" />

<Entry>
  <Title>Catchy Tunes</title>
  <Abstract>Tunes you can't get out of your head</abstract>
  <Copyright>Copyright© 1997 Company Name©</copyright>
  <Ref href = "http://server/catchy.nsc" />
  <Ref href = "http://backup/catchy.nsc" />
</Entry>
</ASX>

```

**Description:** The Entry element is the fundamental construct in an .asx file. The Entry element and its associated attributes define the player meta-information for a single, logical piece of content. This single piece of logical content is called a clip. Elements that are defined within an Entry element are displayed by the player in the Clip information area in the Display panel, as well as the Clip tab in the Properties dialog box.

## EntryRef tag

### Example:

```

<ASX Version = "3.0">
  <Title>Title Goes Here</Title>
  <MoreInfo href = "http://www.server.com" />
  <Entry>
    <Title>Title of This Clip</title>
    <Abstract>More Info About This Clip Goes Here</abstract>
    <Copyright>Copyright 1998 Company Name</copyright>
    <EntryRef href = "http://servername/path/asxname.asx"
      ClientBind = "no" />
  </Entry>
</ASX>

```

**Description:** The EntryRef element is used when the content of an Entry is defined in an external .asx file. Using the EntryRef element is similar to including the textual contents of the referenced .asx file into the current position of the current .asx file.

## Event tag

### Example:

```

<!-- This is a comment section.
The section below defines two dynamic advertisements with dynamic
"meta" data, one regional and one local. When the playing of the ads
is complete the featured content resumes.
End comment section -->

<Event Name = "Regional-Ad" WhenDone = "RESUME" Refresh = "yes">
  <EntryRef href = "regional.asp" ClientSkip = "no" />
</Event>
<Event Name = "Local-Ad" WhenDone = "RESUME" Refresh = "yes">
  <EntryRef href = "local.asp" ClientSkip = "no" />
</Event>

```

**Description:** An event is a script command embedded in a stream sent to the client that consists of a double-string. The first string is the word event (this is case-sensitive), and the second string is the event name. The event name in the second string must match the event name defined in the .asx file. The two are case-sensitive to each other. Events can be sent to a client receiving a real-time stream, or can be saved in an .asf file that gets delivered as an on-demand unicast stream. When the script command is received, the client processes the Event. The Event element

defines a scope of Entry and/or EntryRef elements that play whenever the script command including the named event is received by the player. This allows the author to specify a behavior for stream switching in near real time, as opposed to pre-authored stream changes using references to other pieces of content or .asx files. The preceding example demonstrates using the Event element in an .asx file to deliver advertisements to the client without disconnecting the user from the content being viewed. The author also can define what action the client takes after playing the stream defined in the Event element.

## Logo tag

### Example:

```
<Logo href = "http://server/logos/image1.gif" Style = "ICON" />
<Logo href = "http://server/logos/image1.gif" Style = "IMAGE" />
<Logo href = "http://server/logos/image1.gif" Style = "MARK" />
```

**Description:** The Logo element specifies the URL for a graphic file that is associated with an ASX or Entry element (a show or clip). These graphic files are displayed in the player as they are defined by the Logo element attributes. Logo elements defined within an ASX element (where the ASX element is the parent element) are displayed for the duration of a play list. All graphic styles except “ICON” are displayed in the View area of the player. The “ICON” style images are displayed next to the title of the clip, or shown in the Display panel, depending on the parent of the Logo element. If no “ICON” style Logo elements are included in the ASX, the graphic displayed next to the title is the icon associated with the file name type. Logo formats follow the Microsoft® Internet Explorer Channel guidelines as follows, with all sizes in pixels: ICON - 16 height x 16 width; IMAGE - 32 height x 80 width; MARK - 32 height x 80 width; Windows Media Player supports GIF, BMP, and JPG graphic formats.

## MoreInfo tag

### Example:

```
<!-- This is a comment.
  The MoreInfo tag below makes the Show title in the Display panel
  clickable. If the MoreInfo tag is under an Entry element, the title
  of the clip can be clicked on. When a user places the mouse over the
  title, the URL defined in the href attribute is displayed in the
  Status bar. When a user clicks the title, the browser starts and
  navigates to the page defined in the URL. When the Banner attribute
  is used, the user can click the graphic in the Video area
  to open the URL.
  End comment. -->
<MoreInfo href = "http://www.server.com/info" Banner
  "http://www.server.com/logos/banner.jpg">
</MoreInfo>
<Entry>
  <Ref href = "mms://nsserver/content/title1.asf" />
</Entry>
</ASX>
```

**Description:** The MoreInfo element allows the content creator to provide a URL to a Web site that contains further details about the content the user is viewing (or any other Web site). The URL defined in the href attribute is displayed as a ToolTip when the mouse cursor is placed over the Banner graphic, if one is included. If there is an Abstract element within the scope of the MoreInfo element, the Abstract text is displayed as the ToolTip. When a user clicks on the Banner image, the browser starts and connects to the URL. If there is no Banner attribute in the Logo element, the user accesses the URL from the File menu. On the File menu, there is a More Information submenu that is only available if the open .asx file contains a MoreInfo element. On the More Information submenu, there are three items: Show, Clip, and Banner. If the parent element of the MoreInfo element is an ASX, the Show item is available. When a user clicks Show on the submenu, the browser starts, and opens the page referenced in the href attribute. If the parent element is an Entry, the Clip item is available. If the MoreInfo element contains a Banner

attribute, the Banner item in the More Information submenu is available. Clicking Banner has the same effect as clicking the Banner graphic in the Video area.

## PlayLists tag

### Example:

```
<ASX version = "3.0">
<Title>Most requested titles of the day</Title>
<Entry><Ref href = "mms://nsserver/content/title1.asf" /></Entry>
<Entry><Ref href = "mms://nsserver/content/title2.asf" /></Entry>
<Entry><Ref href = "mms://nsserver/content/title3.asf" /></Entry>
<Entry><Ref href = "mms://nsserver/content/title4.asf" /></Entry>
<ASX>
```

**Description:** A playlist consists of multiple Entry elements in an .asx file. Each Entry element in the .asx file is played by the client in the order they appear in the .asx file as though the user had manually opened each clip.

## PreviewDuration tag

### Example:

```
<Entry>
  <Ref href = "mms://nsserver/content/selection.asf">
    <PreviewDuration value = "0:30.0" />
  </Ref>
</Entry>
```

**Description:** The PreviewDuration element defines the length of time the clip defined in the associated Entry or Ref element is played in Preview Mode. An associated element is the element parent element.

## Ref tag

### Example:

```
<!-- This is a comment.
  There are multiple Ref elements below. Only one clip (or
  station if using .nsc files) will be connected to and played.
  If the URL in the first Ref element fails, the second
  Ref URL is tried. If it also fails, the third Ref URL
  is tried. If all three fail, the next Entry or EntryRef
  element in the ASX, if any, is processed.
End comment. -->
<Entry>
<Ref href = "mms://nsserver/pubpt/selection1.asf" />
<Ref href = "mms://nsserver2/pubpt/selection1.asf" />
<Ref href = "mms://nsbackup/pubpt/selection1.asf" />
</Entry>
```

**Description:** The Ref element specifies a Uniform Resource Locator (URL) for a content stream. The URL can point to either an Advanced Streaming Format (ASF or .asf) file for on-demand content, or an .nsc file that provides the player with information about a Windows Media station. A Windows Media station is a real-time stream. The Ref element can appear only as a child element of an Entry element.

## RefBase tag

### Example:

```
<RefBase href = "http://www.server.com/ads/" />
```

**Description:** The Uniform Resource Locator (URL) defined in the href attribute of the RefBase element is appended to (in front of) the value of the href attribute of both EntryRef and Event elements. The purpose of the RefBase element is to allow the ASX author to use relative links to content rather than specifying the entire path. Href attributes in other elements that contain a protocol (mms:// or http://) are not affected by the RefBase element.

## Repeat tag

### Example:

```
<Repeat Count = "2">
  <Entry><Ref href = "mms://nsserver/clips/clip2.asf" /></Entry>
  <Entry><Ref href = "mms://nsserver/clips/clip3.asf" /></Entry>
</Repeat>
```

**Description:** The Repeat element is used to define the number of times the client repeats the play of an Entry element or set of Entry elements.

## StartMarker tag

### Example:

```
<StartMarker Number = "5" />
<StartMarker Name = "MarkerC" />
```

**Description:** The StartMarker element identifies the named or numbered index marker where the Windows Media Player is to start playing the stream referenced in the associated Entry or Ref element. When the content referenced is opened, it begins playing at the defined marker rather than the beginning of the clip. Markers are present only in stored content.

## StartTime tag

### Example:

```
<Entry>
  <!-- This is a comment.
  The StartTime element below causes the client to begin playing
  the clip one minute and 30 seconds after the beginning.
  End comment. -->
  <StartTime Value = "1:30.0" />
  <Ref href = "mms://nsserver/content/title1.asf" />
</Entry>
```

**Description:** The StartTime element defines a time index into the content where the client is to start playing the stream. The StartTime element can be used only with stored, on-demand content that has been indexed.

## Title tag

### Example:

```
<ASX version = "3.0">
<Title>Title of the Show</title>
<Author>Person who wrote this ASX</author>
<Entry>
  <Title>Title of the Clip</Title>
  <Ref href = "mms://nsserver/content/title1.asf" />
```

</Entry>  
</ASX>

**Description:** The Title element defines a text string displayed by Windows Media Player as the Show or Clip title, depending on the parent element of the Title element. If the parent element is ASX, the text is displayed as the Show title in the Display panel, as well as on the Show tab of the Properties dialog box. If the parent element is Entry, the text is displayed as the Clip title in the Display panel and on the Clip tab of the Properties dialog box. If a MoreInfo element is also used with the associated (parent) element, it is the Title text that the user clicks on to start a browser and navigate to the URL defined in the MoreInfo element.

## ASX Files for Multicast

There are two types of ASX files: handcrafted as discussed earlier, and machine-generated. The machine-generated files are encrypted and they will allow you to publish an announcement that will support multicast. They are generated in [Windows Media Technologies tools](#).

---

[⬆ Back to top](#)

*Did you find this material useful? Gripes? Compliments? Suggestions for other articles? [Write us!](#)*

© [1999 Microsoft Corporation. All rights reserved. Terms of use.](#)



# Synchronized Multimedia Integration Language (SMIL) 1.0 Specification

W3C Recommendation 15-June-1998

This version:

<http://www.w3.org/TR/1998/REC-smil-19980615>

Latest version:

<http://www.w3.org/TR/REC-smil>

Previous version:

<http://www.w3.org/TR/1998/PR-smil-19980409>

Copyright © 1998 W3C (MIT, INRIA, Keio), All Rights Reserved. W3C [liability](#), [trademark](#), [document use](#) and [software licensing](#) rules apply. Your interactions with this site are in accordance with our [public](#) and [Member](#) privacy statements.

## About this Document

This document has been prepared by the Synchronized Multimedia Working Group (WG) of the World Wide Web Consortium. The WG included the following individuals:

- Stephan Bugaj, Lucent/Bell Labs
- Dick Bulterman, CWI
- Bruce Butterfield, RealNetworks
- Wo Chang, NIST
- Guy Fouquet, Alcatel
- Christian Gran, GMD
- Mark Hakkinen, The Productivity Works
- Lynda Hardman, CWI
- Peter Hoddie, Apple
- Klaus Hofrichter, GMD
- Philipp Hoschka, W3C
- Jack Jansen, CWI
- George Kerscher, DAISY Consortium
- Rob Lanphier, RealNetworks
- Nabil Layaïda, INRIA
- Stephanie Leif, RealNetworks
- Sjoerd Mullender, CWI
- Didier Pillet, CNET/DSM
- Anup Rao, Netscape
- Lloyd Rutledge, CWI
- Patrick Soquet, Havas
- Warner ten Kate, Philips

- Jacco van Ossenbruggen, CWI
- Michael Vernick, Lucent/Bell Labs
- Jin Yu, DEC

**Acknowledgments:** In addition to the working group members, the following people contributed to the SMIL effort: Bert Bos (W3C), Dan Connolly (W3C), Patrick Deunhouwer (Philips), Martin Dürst (W3C), Al Gilman, Håkon Lie (W3C), Chris Lilley (W3C), Curtis Reynolds (RealNetworks), Michael Riesman, Curtis Reynolds (RealNetworks), Henning Schulzrinne (Columbia University) and Koga Youichirou (W3C).

**Editor:** Philipp Hoschka, W3C ([hoschka@w3.org](mailto:hoschka@w3.org))

## Abstract

This document specifies version 1 of the Synchronized Multimedia Integration Language (SMIL 1.0, pronounced "smile"). SMIL allows integrating a set of independent multimedia objects into a synchronized multimedia presentation. Using SMIL, an author can

1. describe the temporal behavior of the presentation
2. describe the layout of the presentation on a screen
3. associate hyperlinks with media objects

This specification is structured as follows: Section 1 presents the specification approach. Section 2 defines the "smil" element. Section 3 defines the elements that can be contained in the head part of a SMIL document. Section 4 defines the elements that can be contained in the body part of a SMIL document. In particular, this Section defines the time model used in SMIL. Section 5 describes the SMIL DTD.

## Status of this Document

This document has been reviewed by W3C Members and other interested parties and has been endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited as a normative reference from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

Comments on this Recommendation may be sent to the [public mailing list www-smil@w3.org](mailto:public-mailing-list-www-smil@w3.org).

## Available languages

The English version of this specification is the only normative version. However, for translations in other languages see <http://www.w3.org/AudioVideo/SMIL/translations>.

## Errata

The list of known errors in this specification is available at <http://www.w3.org/AudioVideo/SMIL/errata>.

## Table of Contents

- [1 Specification Approach](#)
- [2 The smil Element](#)
- [3 The Document Head](#)
  - [3.1 The head Element](#)



- [3.2 The layout Element](#)
- [3.3 SMIL Basic Layout Language](#)
  - [3.3.1 The region Element](#)
  - [3.3.2 The root-layout Element](#)
- [3.4 The meta Element](#)
- [4 The Document Body](#)
  - [4.1 The body Element](#)
  - [4.2 Synchronization Elements](#)
    - [4.2.1 The par Element](#)
    - [4.2.2 The seq Element](#)
    - [4.2.3 Media Object Element](#): The ref, animation, audio, img, video, text and textstream elements
    - [4.2.4 SMIL Time Model](#)
      - [4.2.4.1 Time Model Values](#)
      - [4.2.4.2 Determining Values of Model Values for SMIL 1.0 Elements](#)
  - [4.3 The switch Element](#)
  - [4.4 Test Attributes](#)
  - [4.5 Hyperlinking Elements](#)
    - [4.5.1 The a Element](#)
    - [4.5.2 The anchor Element](#)
- [5 SMIL DTD](#)
  - [5.1 Relation to XML](#)
  - [5.2 DTD](#)
- [Appendix](#)
  - [Extending SMIL 1.0](#)
  - [Using SMIL 1.0 as an Extension](#)

## 1 Specification Approach

SMIL documents are XML 1.0 documents [\[XML10\]](#). The reader is expected to be familiar with the concepts and terms defined in XML 1.0.

This specification does not rely on particular features defined in URLs that cannot potentially be expressed using URNs. Therefore, the more generic term URI [\[URI\]](#) is used throughout the specification.

The syntax of SMIL documents is defined by the DTD in [Section 5.2](#). The syntax of an attribute value that cannot be defined using the DTD notation is defined together with the first element using an attribute that can contain the attribute value. The syntax of such attribute values is defined using the Extended Backus-Naur Form (EBNF) defined in the XML 1.0 specification.

An element definition is structured as follows: First, all attributes of the element are defined in alphabetical order. An attribute is defined in the following way: If the attribute is used by an element for the first time in the specification, the semantics of the attribute are defined. If the attribute has already been used by another element, the specification refers to the definition of the attribute in the first element that used it. The definition of element attributes is followed by the definition of any attribute values whose syntax cannot be defined using the DTD notation. The final section in an element definition specifies the element content.

## 2 The `smil` Element

### Element Attributes

The "smil" element can have the following attribute:

id

This attribute uniquely identifies an element within a document. Its value is an XML identifier.

### Element Content

The "smil" element can contain the following children:

body

Defined in [Section 4.1](#)

head

Defined in [Section 3.1](#)

## 3 The Document Head

### 3.1 The head Element

The "head" element contains information that is not related to the temporal behavior of the presentation.

#### Element Attributes

The "head" element can have the following attribute:

id

Defined in [Section 2](#)

#### Element Content

The "head" element can contain the following children:

layout

Defined in [Section 3.2](#)

meta

Defined in [Section 3.4](#)

switch

Defined in [Section 4.3](#)

The "head" element may contain any number of "meta" elements and either a "layout" element or a "switch" element.

### 3.2 The layout Element

The "layout" element determines how the elements in the document's body are positioned on an abstract rendering surface (either visual or acoustic).

If a document contains no layout element, the positioning of the body elements is implementation-dependent.

A SMIL document can contain multiple alternative layouts by enclosing several layout elements within a "switch" element (defined in [Section 4.3](#)). This can be used for example to describe the document's layout using different layout languages.

The following example shows how CSS2 can be used as alternative to the SMIL basic layout language (defined in [Section 3.3](#)):

```
<smil>  
<head>
```

```

<switch>
  <layout type="text/css">
    [region="r"] { top: 20px; left: 20px }
  </layout>
</layout>
  <region id="r" top="20" left="20" />
</layout>
</switch>
</head>
<body>
  <seq>
    
  </seq>
</body>
</smil>

```

(note that in this example, both layout alternatives result in the same layout)

## Element Attributes

**id**  
Defined in [Section 2](#)

**type**  
This attribute specifies which layout language is used in the layout element. If the player does not understand this language, it must skip all content up until the next "</layout>" tag. The default value of the type attribute is "text/smil-basic-layout".

## Element Content

If the type attribute of the layout element has the value "text/smil-basic-layout", it can contain the following elements:

**region**  
Defined in [Section 3.3.1](#)

**root-layout**  
Defined in [Section 3.3.2](#)

If the type attribute of the "layout" element has another value, the element contains character data.

## 3.3 SMIL Basic Layout Language

This section defines a basic layout language for SMIL. SMIL basic layout is consistent with the visual rendering model defined in CSS2, it reuses the formatting properties defined by the CSS2 specification, and newly introduces the "fit" attribute [\[CSS2\]](#). The reader is expected to be familiar with the concepts and terms defined in CSS2.

SMIL basic layout only controls the layout of media object elements (defined in [Section 4.2.3](#)). It is illegal to use SMIL basic layout for other SMIL elements.

The type identifier for SMIL basic layout is "text/smil-basic-layout".

## Fixed Property Values

The following stylesheet defines the values of the CSS2 properties "display" and "position" that are valid in SMIL basic layout. These property values are fixed:

```

a          {display:block}
anchor     {display:block}

```

```

animation    {display: block;
               position: absolute}
body         {display: block}
head         {display: none}
img          {display: block;
               position: absolute}
layout       {display: none}
meta         {display: none}
par          {display: block}
region       {display: none}
ref          {display: block;
               position: absolute}
root-layout  {display: none}
seq          {display: block}
smil         {display: block}
switch       {display: block}
text         {display: block;
               position: absolute}
textstream   {display: block;
               position: absolute}
video        {display: block;
               position: absolute}

```

Note that as a result of these definitions, all absolutely positioned elements (animation, img, ref, text, textstream and video) are contained within a single containing block defined by the content content edge of the root element (smil).

## Default Values

SMIL basic layout defines default values for all layout-related attributes. These are consistent with the initial values of the corresponding properties in CSS2.

If the author wants to select the default layout values for *all* media object elements in a document, the document must contain an empty layout element of type "text/smil-basic-layout" such as:

```
<layout type="text/smil-basic-layout"></layout>
```

### 3.3.1 The region Element

The region element controls the position, size and scaling of media object elements.

In the following example fragment, the position of a text element is set to a 5 pixel distance from the top border of the rendering window:

```

<smil>
  <head>
    <layout>
      <region id="a" top="5" />
    </layout>
  </head>
  <body>
    <text region="a" src="text.html" dur="10s" />
  </body>
</smil>

```

## Element Attributes

The "region" element can have the following attributes:

background-color

The use and definition of this attribute are identical to the "background-color" property in the CSS2 specification, except that SMIL basic layout does not require support for "system colors".

If the background-color attribute is absent, the background is transparent.

fit

This attribute specifies the behavior if the intrinsic height and width of a visual media object differ from the values specified by the height and width attributes in the "region" element. This attribute does not have a 1-1 mapping onto a CSS2 property, but can be simulated in CSS2.

This attribute can have the following values:

fill

Scale the object's height and width independently so that the content just touches all edges of the box.

hidden

- If the intrinsic height (width) of the media object element is smaller than the height (width) defined in the "region" element, render the object starting from the top (left) edge and fill up the remaining height (width) with the background color.
- If the intrinsic height (width) of the media object element is greater than the height (width) defined in the "region" element, render the object starting from the top (left) edge until the height (width) defined in the "region" element is reached, and clip the parts of the object below (right of) the height (width).

meet

Scale the visual media object while preserving its aspect ratio until its height or width is equal to the value specified by the height or width attributes, while none of the content is clipped. The object's left top corner is positioned at the top-left coordinates of the box, and empty space at the left or bottom is filled up with the background color.

scroll

A scrolling mechanism should be invoked when the element's rendered contents exceed its bounds.

slice

Scale the visual media object while preserving its aspect ratio so that its height or width are equal to the value specified by the height and width attributes while some of the content may get clipped. Depending on the exact situation, either a horizontal or a vertical slice of the visual media object is displayed. Overflow width is clipped from the right of the media object. Overflow height is clipped from the bottom of the media object.

The default value of "fill" is "hidden".

height

The use and definition of this attribute are identical to the "height" property in the CSS2 specification. Attribute values can be "percentage" values, and a variation of the "length" values defined in CSS2. For "length" values, SMIL basic layout only supports pixel units as defined in CSS2. It allows to leave out the "px" unit qualifier in pixel values (the "px" qualifier is required in CSS2).

id

Defined in [Section 2](#)

A region element is applied to a positionable element by setting the [region](#) attribute of the positionable element to the id value of the region.

The "id" attribute is required for "region" elements.

left

The use and definition of this attribute are identical to the "left" property in the CSS2 specification. Attribute values have the same restrictions as the attribute values of the "height" attribute.

The default value is zero.

skip-content

This attribute is introduced for future extensibility of SMIL (see [Appendix](#)). It is interpreted in the following two cases:

- If a new element is introduced in a future version of SMIL, and this element allows SMIL 1.0 elements as element content, the "skip-content" attribute controls whether this content is processed by a SMIL 1.0 player.
- If an empty element in SMIL version 1.0 becomes non-empty in a future SMIL version, the "skip-content" attribute controls whether this content is ignored by a SMIL 1.0 player, or results in a syntax error.

If the value of the "skip-content" attribute is "true", and one of the cases above apply, the content of the element is ignored. If the value is "false", the content of the element is processed.

The default value for "skip-content" is "true".

#### title

This attribute offers advisory information about the element for which it is set. Values of the title attribute may be rendered by user agents in a variety of ways. For instance, visual browsers frequently display the title as a "tool tip" (a short message that appears when the pointing device pauses over an object).

It is strongly recommended that all "region" elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

#### top

The use and definition of this attribute are identical to the "top" property in the CSS2 specification.

Attribute values have the same restrictions as the attribute values of the "height" attribute.

The default value is zero.

#### width

The use and definition of this attribute are identical to the "width" property in the CSS2 specification.

Attribute values have the same restrictions as the attribute values of the "height" attribute.

#### z-index

The use and definition of this attribute are identical to the "z-index" property in the CSS2 specification, with the following exception:

- If two boxes generated by elements A and B have the same stack level, then
  1. If the display of an element A starts later than the display of an element B, the box of A is stacked on top of the box of B (temporal order).
  2. If the display of the elements starts at the same time, and an element A occurs later in the SMIL document text than an element B, the box of A is stacked on top of the box of B (document tree order as defined in CSS2).

### Element Content

"region" is an empty element.

### 3.3.2 The root-layout element

The "root-layout" element determines the value of the layout properties of the root element, which in turn determines the size of the viewport, e.g. the window in which the SMIL presentation is rendered.

If a document contains more than one "root-layout" element, this is an error, and the document should not be displayed.

### Element Attributes

The "root-layout" element can have the following attributes:

background-color

Defined in [Section 3.3.1](#)

height

Defined in [Section 3.3.1](#)

Sets the height of the root element. Only length values are allowed.

id

Defined in [Section 2](#)

skip-content

Defined in [Section 3.3.1](#)

title

Defined in [Section 3.3.1](#)

width

Defined in [Section 3.3.1](#)

Sets the width of the root element. Only length values are allowed.

## Element Content

"root-layout" is an empty element.

## 3.4 The meta Element

The "meta" element can be used to define properties of a document (e.g., author, expiration date, a list of key words, etc.) and assign values to those properties. Each "meta" element specifies a single property/value pair.

### Element Attributes

The "meta" element can have the following attributes:

content

This attribute specifies the value of the property defined in the meta element.

The "content" attribute is required for "meta" elements.

id

Defined in [Section 2](#)

name

This attribute identifies the property defined in the meta element.

The "name" attribute is required for "meta" elements.

skip-content

Defined in [Section 3.3.1](#)

The list of properties is open-ended. This specification defines the following properties:

base

The value of this property determines the base URI for all relative URIs used in the document.

pics-label or PICS-Label

The value of this property specifies a valid rating label for the document as defined by PICS [\[PICS\]](#).

title

The value of this property contains the title of the presentation.

### Element Content

"meta" is an empty element.

## 4 The Document Body

### 4.1 The body Element

The "body" element contains information that is related to the temporal and linking behavior of the document. It implicitly defines a "seq" element (defined in Section 4.2.2, see Section 4.2.4 for a definition of the temporal semantics of the "body" element).

### Element Attributes

The "body" element can have the following attribute:

id  
Defined in [Section 2](#)

### Element Content

The "body" element can contain the following children:

a  
Defined in [Section 4.5.1](#)  
animation  
Defined in [Section 4.2.3](#)  
audio  
Defined in [Section 4.2.3](#)  
img  
Defined in [Section 4.2.3](#)  
par  
Defined in [Section 4.2.1](#)  
ref  
Defined in [Section 4.2.3](#)  
seq  
Defined in [Section 4.2.2](#)  
switch  
Defined in [Section 4.3](#)  
text  
Defined in [Section 4.2.3](#)  
textstream  
Defined in [Section 4.2.3](#)  
video  
Defined in [Section 4.2.3](#)

## 4.2 Synchronization Elements

### 4.2.1 The par Element

The children of a par element can overlap in time. The textual order of appearance of children in a par has no significance for the timing of their presentation.

### Element Attributes

The "par" element can have the following attributes:

abstract  
A brief description of the content contained in the element.  
author  
The name of the author of the content contained in the element.  
begin



This attribute specifies the time for the explicit begin of an element. See [Section 4.2.4](#) for a definition of its semantics.

The attribute can contain the following two types of values:

#### delay-value

A delay value is a clock-value measuring presentation time. Presentation time advances at the speed of the presentation. It behaves like the timecode shown on a counter of a tape-deck. It can be stopped, decreased or increased either by user actions, or by the player itself.

The semantics of a delay value depend on the element's first ancestor that is a synchronization element (i.e. ancestors that are "a" or "switch" elements are ignored):

- If this ancestor is a "par" element, the value defines a delay from the effective begin of that element (see Figure 4.1).
- If this ancestor is a "seq" element (defined in [Section 4.2.2](#)), the value defines a delay from the effective end of the first lexical predecessor that is a synchronization element (see Figure 4.2).

#### event-value

The element begins when a certain event occurs (see Figure 4.3). Its value is an element-event (see Definition below).

The element generating the event must be "in scope". The set of "in scope" elements S is determined as follows:

1. Take all children from the element's first ancestor that is a synchronization element and add them to S.
2. Remove all "a" and "switch" elements from S. Add the children of all "a" elements to S, unless they are "switch" elements.

The resulting set S is the set of "in scope" elements.

---

```
<par>
  <audio id="a" begin="6s" src="audio" />
</par>
```

[D](#)

---

Figure 4.1: Using a delay value within a "par" element

---

```
<seq>
  <audio src="audiol" />
  <audio begin="5s" src="audio2" />
</seq>
```

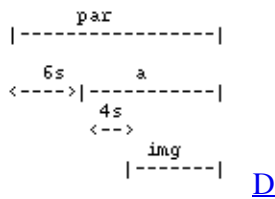
[D](#)

---

Figure 4.2: Using a delay value within a "seq" element

---

```
<par>
  <audio id="a" begin="6s" ... />
  <img begin="id(a)(4s)" ... />
</par>
```



D

Figure 4.3: Synchronization attribute with element event value

#### copyright

The copyright notice of the content contained in the element.

#### dur

This attribute specifies the explicit duration of an element. See [Section 4.2.4](#) for a definition of its semantics. The attribute value can be a clock value, or the string "indefinite".

#### end

This attribute specifies the explicit end of an element. See [Section 4.2.4](#) for a definition of its semantics. The attribute can contain the same types of attribute values as the "begin" attribute.

#### endsync

For a definition of the semantics of this attribute, see [Section 4.2.4](#). The attribute can have the following values:

- first  
For a definition of the semantics of this value, see [Section 4.2.4](#).
- id-ref  
This attribute value has the following syntax:  
  

```
id-ref ::= "id(" id-value ")"
```

 where "id-value" must be a legal XML identifier.  
For a definition of the semantics of this value, see [Section 4.2.4](#).
- last  
For a definition of the semantics of this value, see [Section 4.2.4](#).

The default value of "endsync" is "last".

#### id

Defined in [Section 2](#)

#### region

This attribute specifies an abstract rendering surface (either visual or acoustic) defined within the layout section of the document. Its value must be an XML identifier. If no rendering surface with this id is defined in the layout section, the values of the formatting properties of this element are determined by the default layout.

The "region" attribute on "par" elements cannot be used by the basic layout language for SMIL defined in this specification. It is added for completeness, since it may be required by other layout languages.

#### repeat

For a definition of the semantics of this attribute, see [Section 4.2.4](#). The attribute value can be an integer, or the string "indefinite". The default value is 1.

#### system-bitrate

Defined in [Section 4.4](#)

#### system-captions

Defined in [Section 4.4](#)

#### system-language

Defined in [Section 4.4](#)

#### system-overdub-or-caption

Defined in [Section 4.4](#)

system-required

Defined in [Section 4.4](#)

system-screen-size

Defined in [Section 4.4](#)

system-screen-depth

Defined in [Section 4.4](#)

title

Defined in [Section 3.3.1](#)

It is strongly recommended that all "par" elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

### Note on Synchronization between Children

The accuracy of synchronization between the children in a parallel group is implementation-dependent. Take the example of synchronization in case of playback delays, i.e. the behavior when the "par" element contains two or more continuous media types such as audio or video, and one of them experiences a delay.

A player can show the following synchronization behaviors:

hard synchronization

The player synchronizes the children in the "par" element to a common clock (see Figure 4.4 a)).

soft synchronization

Each child of the "par" element has its own clock, which runs independently of the clocks of other children in the "par" element (see Figure 4.4 b)).

---

```

audio
|-----|
video
|-----|

```

```

audio
|-----|
video
|-----|

```

[D](#)

a) *hard synchronization: Delay in video: Either the audio is stopped, or some video frames are dropped. The exact behavior is implementation-dependent*

```

audio
|-----|
video
|-----|

```

[D](#)

b) *soft synchronization*

---

Figure 4.4: Effect of a delay on playout schedule for players using different synchronization policies

### Attribute Values

clock value

Clock values have the following syntax:

```

Clock-val      ::= Full-clock-val | Partial-clock-val | Timecount-val
Full-clock-val ::= Hours ":" Minutes ":" Seconds ( "." Fraction )?
Partial-clock-val ::= Minutes ":" Seconds ( "." Fraction )?
Timecount-val  ::= Timecount ( "." Fraction )?
                  ( "h" | "min" | "s" | "ms" )? ; default is "s"
Hours          ::= 2DIGIT; any positive number

```

Minutes	::= 2DIGIT; range from 00 to 59
Seconds	::= 2DIGIT; range from 00 to 59
Fraction	::= DIGIT+
Timecount	::= DIGIT+
2DIGIT	::= DIGIT DIGIT
DIGIT	::= [0-9]

The following are examples of legal clock values:

- Full clock value: 02:30:03 = 2 hours, 30 minutes and 3 seconds
- Partial clock value: 02:33 = 2 minutes and 33 seconds
- Timecount values:
  - 3h = 3 hours
  - 45min = 45 minutes
  - 30s = 30 seconds
  - 5ms = 5 milliseconds

A fraction  $x$  with  $n$  digits represents the following value:

$$x * 1/10^{**n}$$

Examples:

00.5s =  $5 * 1/10$  seconds = 500 milliseconds

00:00.005 =  $5 * 1/1000$  seconds = 5 milliseconds

#### element-event value

An *element event* value specifies a particular event in a synchronization element.

An element event has the following syntax:

Element-event	::= "id(" Event-source ") (" Event ") "
Event-source	::= Id-value
Event	::= "begin"   Clock-val   "end"

The following events are defined:

#### begin

This event is generated at an element's effective begin.

Example use: begin="id(x)(begin) "

#### clock-val

This event is generated when a clock associated with an element reaches a particular value. This clock starts at 0 at the element's effective begin. For "par" and "seq" elements, the clock gives the presentation time elapsed since the effective begin of the element. For media object elements, the semantics are implementation-dependent. The clock may either give presentation time elapsed since the effective begin, or it may give the media time of the object. The latter may differ from the presentation time that elapsed since the object's display was started e.g. due to rendering or network delays, and is the recommended approach.

It is an error to use a clock value that exceeds the value of the effective duration of the element generating the event.

Example use: begin="id(x)(45s) "

#### end

This event is generated at the element's effective end.

Example use: `begin="id(x)(end)"`

## Element Content

The par element can contain the following children:

a	Defined in <a href="#">Section 4.5.1</a>
animation	Defined in <a href="#">Section 4.2.3</a>
audio	Defined in <a href="#">Section 4.2.3</a>
img	Defined in <a href="#">Section 4.2.3</a>
par	Defined in <a href="#">Section 4.2.1</a>
ref	Defined in <a href="#">Section 4.2.3</a>
seq	Defined in <a href="#">Section 4.2.2</a>
switch	Defined in <a href="#">Section 4.3</a>
text	Defined in <a href="#">Section 4.2.3</a>
textstream	Defined in <a href="#">Section 4.2.3</a>
video	Defined in <a href="#">Section 4.2.3</a>

All of these elements may appear multiple times as direct children of a par element.

### 4.2.2 The seq Element

The children of a "seq" element form a temporal sequence.

#### Attributes

The seq element can have the following attributes:

abstract	Defined in <a href="#">Section 4.2.1</a>
author	Defined in <a href="#">Section 4.2.1</a>
begin	Defined in <a href="#">Section 4.2.1</a>
copyright	Defined in <a href="#">Section 4.2.1</a>
dur	Defined in <a href="#">Section 4.2.1</a>
end	Defined in <a href="#">Section 4.2.1</a>
id	Defined in <a href="#">Section 2</a>

**region**Defined in [Section 4.2.1](#)

The region attribute on "seq" elements cannot be used by the basic layout language for SMIL defined in this specification. It is added for completeness, since it may be required by other layout languages.

**repeat**Defined in [Section 4.2.1](#)**system-bitrate**Defined in [Section 4.4](#)**system-captions**Defined in [Section 4.4](#)**system-language**Defined in [Section 4.4](#)**system-overdub-or-caption**Defined in [Section 4.4](#)**system-required**Defined in [Section 4.4](#)**system-screen-size**Defined in [Section 4.4](#)**system-screen-depth**Defined in [Section 4.4](#)**title**Defined in [Section 3.3.1](#)

It is strongly recommended that all "seq" elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

**Element Content**

The seq element can contain the following children:

**a**Defined in [Section 4.5.1](#)**animation**Defined in [Section 4.2.3](#)**audio**Defined in [Section 4.2.3](#)**img**Defined in [Section 4.2.3](#)**par**Defined in [Section 4.2.1](#)**ref**Defined in [Section 4.2.3](#)**seq**Defined in [Section 4.2.2](#)**switch**Defined in [Section 4.3](#)**text**Defined in [Section 4.2.3](#)**textstream**Defined in [Section 4.2.3](#)**video**Defined in [Section 4.2.3](#)**4.2.3 Media Object Elements: The ref, animation, audio, img, video, text and textstream elements**

The media object elements allow the inclusion of media objects into a SMIL presentation. Media objects are included by reference (using a URI).

There are two types of media objects: media objects with an intrinsic duration (e.g. video, audio) (also called "continuous media"), and media objects without intrinsic duration (e.g. text, image) (also called "discrete media").

anchors and links can be attached to visual media objects, i.e. media objects rendered on a visual abstract rendering surface.

When playing back a media object, the player must not derive the exact type of the media object from the name of the media object element. Instead, it must rely solely on other sources about the type, such as type information contained in the "type" attribute, or the type information communicated by a server or the operating system.

Authors, however, should make sure that the group into which of the media object falls (animation, audio, img, video, text or textstream) is reflected in the element name. This is in order to increase the readability of the SMIL document. When in doubt about the group of a media object, authors should use the generic "ref" element.

## Element Attributes

Media object elements can have the following attributes:

abstract

Defined in [Section 4.2.1](#)

alt

For user agents that cannot display a particular media-object, this attribute specifies alternate text. It is strongly recommended that all media object elements have an "alt" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

author

Defined in [Section 4.2.1](#)

begin

Defined in [Section 4.2.1](#)

clip-begin

The clip-begin attribute specifies the beginning of a sub-clip of a continuous media object as offset from the start of the media object.

Values in the clip-begin attribute have the following syntax:

```
Clip-time-value ::= Metric "=" ( Clock-val | Smpte-val )
Metric          ::= Smpte-type | "npt"
Smpte-type      ::= "smpte" | "smpte-30-drop" | "smpte-25"
Smpte-val       ::= Hours ":" Minutes ":" Seconds
                  [ ":" Frames [ "." Subframes ] ]
Hours           ::= 2DIGIT
Minutes         ::= 2DIGIT
Seconds         ::= 2DIGIT
Frames          ::= 2DIGIT
Subframes       ::= 2DIGIT
```

The value of this attribute consists of a metric specifier, followed by a time value whose syntax and semantics depend on the metric specifier. The following formats are allowed:

**SMPTE Timestamp**

SMPTE time codes [\[SMPTE\]](#) can be used for frame-level access accuracy. The metric specifier can have the following values:

smpte

**smpte-30-drop**

These values indicate the use of the "SMPTE 30 drop" format with 29.97 frames per second. The "frames" field in the time value can assume the values 0 through 29. The difference between 30 and 29.97 frames per second is handled by dropping the first two frame indices (values 00 and 01) of every minute, except every tenth minute.

**smpte-25**

The "frames" field in the time specification can assume the values 0 through 24.

The time value has the format hours:minutes:seconds:frames.subframes. If the frame value is zero, it may be omitted. Subframes are measured in one-hundredth of a frame.

Examples:

```
clip-begin="smpte=10:12:33:20"
```

**Normal Play Time**

Normal Play Time expresses time in terms of SMIL clock values. The metric specifier is "npt", and the syntax of the time value is identical to the syntax of SMIL clock values.

Examples:

```
clip-begin="npt=123.45s"
```

```
clip-begin="npt=12:05:35.3"
```

**clip-end**

The clip-end attribute specifies the end of a sub-clip of a continuous media object (such as audio, video or another presentation) that should be played. It uses the same attribute value syntax as the clip-begin attribute.

If the value of the "clip-end" attribute exceeds the duration of the media object, the value is ignored, and the clip end is set equal to the effective end of the media object.

**copyright**

Defined in [Section 4.2.1](#)

**dur**

Defined in [Section 4.2.1](#)

**end**

Defined in [Section 4.2.1](#)

**fill**

For a definition of the semantics of this attribute, see Section 4.2.4. The attribute can have the values "remove" and "freeze".

**id**

Defined in [Section 2](#)

**longdesc**

This attribute specifies a link (URI) to a long description of a media object. This description should supplement the short description provided using the alt attribute. When the media-object has associated anchors, this attribute should provide information about the anchor's contents.

**region**

Defined in [Section 4.2.1](#)

**src**

The value of the src attribute is the URI of the media object.

**system-bitrate**

Defined in [Section 4.4](#)

**system-captions**

Defined in [Section 4.4](#)

**system-language**

Defined in [Section 4.4](#)

**system-overdub-or-caption**

Defined in [Section 4.4](#)

**system-required**

Defined in [Section 4.4](#)



system-screen-size

Defined in [Section 4.4](#)

system-screen-depth

Defined in [Section 4.4](#)

title

Defined in [Section 3.3.1](#)

It is strongly recommended that all media object elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

type

MIME type of the media object referenced by the "src" attribute.

## Element Content

Media Object Elements can contain the following element:

anchor

Defined in [Section 4.5.2](#)

## 4.2.4 SMIL Time Model

### 4.2.4.1 Time Model Values

In the following discussion, the term "element" refers to synchronization elements only.

For each element we define the implicit, explicit, desired, and effective begin, duration, and end.

The effective begin/duration/end specify what the reader of the document will perceive.

The implicit, explicit, and desired values are auxiliary values used to define the effective values.

The rules for calculating each of these values for the elements defined in SMIL 1.0 are described in the next section.

1. Each element in SMIL has an *implicit begin*.
2. Each element can be assigned an *explicit begin* by adding a "begin" attribute to the element:

`begin = "value of explicit begin"`

It is an error if the explicit begin is earlier than the implicit begin of the element.

3. Each element in SMIL has an *implicit end*.
4. Each element can be assigned an *explicit end* by adding an "end" attribute to the element:

`end = "value of explicit end"`

5. The *implicit duration* of an element is the difference between the implicit end and the implicit begin.
6. Each element in SMIL can be assigned an *explicit duration* by adding a "dur" attribute to the element:

`dur = "value of explicit duration"`

7. The *desired begin* of an element is equal to the explicit begin if one is given, otherwise the desired begin is equal to the implicit begin.
8. Each element has a *desired end*.
9. The *desired duration* of an element is the difference between the desired end and the desired begin.
10. Each element has an *effective begin*.

11. Each element has an *effective end*. (Note: the effective end of a child element can never be later than the effective end of its parent.)
12. The *effective duration* of an element is the difference between the effective end and the effective begin.

#### 4.2.4.2 Determining Time Model Values for SMIL 1.0 Elements

This section defines how time model values are calculated for the synchronization elements of SMIL 1.0 in cases that are not covered by the rules in [Section 4.2.4.1](#).

##### Determining the *implicit begin* of an element

- The implicit begin of the first child of the "body" element is when the document starts playing. When this is falls outside the scope of this document.
- The implicit begin of a child of a "par" element is equal to the effective begin of the "par" element.
- The implicit begin of the first child of a "seq" element is equal to the effective begin of the "seq" element.
- The implicit begin of any other child of a "seq" element is equal to the desired end time of the previous child of the "seq" element.

##### Determining the *implicit end* of an element

The first description that matches the element is the one that is to be used:

- An element with a "repeat" attribute with value "indefinite" has an implicit end immediately after its effective begin.
- An element with a "repeat" attribute with a value other than "indefinite" has an implicit end equal to the implicit end of a seq element with the stated number of copies of the element without "repeat" attribute as children.
- A media object element referring to a continuous media object has an implicit end equal to the sum of the effective begin of the element and the intrinsic duration of the media object.
- A media object element referring to a discrete media object such as text or image has an implicit end immediately after its effective begin.
- A "seq" element has an implicit end equal to the desired end of its last child.
- A "par" element has an implicit end that depends on the value of the "endsync" attribute. The implicit end is equal to the sum of the effective begin of the "par" element and the implicit duration which is derived as follows:
  - If the value of the "endsync" attribute is "last", or if the "endsync" attribute is missing, the implicit duration of the "par" element is the maximum of the desired durations of its children.
  - If the value of the "endsync" attribute is "first", the implicit duration of the "par" element is the minimum of the desired durations of its children.
  - If the value of the "endsync" attribute is an id-ref, the implicit duration of the "par" element is equal to the desired duration of the child referenced by the "id-ref".

##### Determining the *desired end* of an element

- If the element has both an explicit duration and an explicit end, the desired end is the minimum of:
  - the sum of the desired begin and the explicit duration; and
  - the explicit end.
- If the element has an explicit duration but no explicit end, the desired end is the sum of the desired begin and the explicit duration.
- If the element has an explicit end but no explicit duration, the desired end is equal to the explicit end
- Otherwise, the desired end is equal to the implicit end.

##### Determining the *desired begin* of an element

The desired begin of an element is determined by using rule 7 in [Section 4.2.4.1](#).

### Determining the *effective begin* of an element

The *effective begin* of an element is equal to the desired begin of the element, unless the effective end of the parent element is earlier than this time, in which case the element is not shown at all.

### Determining the *effective end* of an element

- The effective end of the last child of the body element is player-dependent. The effective end is at least as late as the desired end, but whether it is any later is implementation-dependent.
- The effective end of the child of a "par" element can be derived as follows:
  - If the child has a "fill" attribute, and the value of the "fill" attribute is "freeze", the effective end of the child element is equal to the effective end of the parent.  
The last state of the element is retained on the screen until the effective end of the element.
  - If the child has a "fill" attribute, and the value of the "fill" attribute is "remove", the effective end of the child element is the minimum of the effective end of the parent and the desired end of the child element.
  - If the child element has no "fill" attribute, the effective end of the child depends on whether or not the child has an explicit duration or end.
    - If the child has an explicit duration or end, the effective end is determined as if the element had a "fill" attribute with value "remove".
    - If the child has neither an explicit duration nor an explicit end, the effective end is determined as if the element had a "fill" attribute with value "freeze".
- The effective end of the last child of a "seq" element is derived in the same way as the effective end of a child of a "par" element.
- The effective end of any other child of a "seq" element can be derived as follows:
  - If the child has a "fill" attribute, and the value of the "fill" attribute is "freeze", the effective end of the child element is equal to the effective begin of the next element
  - If the child has a "fill" attribute, and the value of the "fill" attribute is "remove", the effective end of the child element is the minimum of the effective begin of the next element and the desired end of the next child element.
  - If the child element has no "fill" attribute, the effective end of the child depends on whether or not the child has an explicit duration or end.
    - If the child has an explicit duration or end, the effective end is determined as if the element had a fill attribute with value "remove".
    - If the child has neither an explicit duration nor an explicit end, the effective end is determined as if the element had a fill attribute with value "freeze".

## 4.3 The switch Element

The switch element allows an author to specify a set of alternative elements from which only one acceptable element should be chosen. An element is acceptable if the element is a SMIL 1.0 element, the media-type can be decoded, and all of the test-attributes (see [Section 4.4](#)) of the element evaluate to "true".

An element is selected as follows: the player evaluates the elements in the order in which they occur in the switch element. The first acceptable element is selected at the exclusion of all other elements within the switch.

Thus, authors should order the alternatives from the most desirable to the least desirable. Furthermore, authors should place a relatively fail-safe alternative as the last item in the <switch> so that at least one item within the switch is chosen (unless this is explicitly not desired). Implementations should NOT arbitrarily pick an object within a <switch> when test-attributes for all fail.

Note that http URIs provide for content-negotiation, which may be an alternative to using the "switch" element in some cases.

## Attributes

The switch element can have the following attributes:

id	Defined in <a href="#">Section 2</a>
title	Defined in <a href="#">Section 3.3.1</a> It is strongly recommended that all switch elements have a "title" attribute with a meaningful description Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

## Element Content

If the "switch" element is used as a direct or indirect child of a "body" element, it can contain the following children:

a	Defined in <a href="#">Section 4.5.1</a>
animation	Defined in <a href="#">Section 4.2.3</a>
audio	Defined in <a href="#">Section 4.2.3</a>
img	Defined in <a href="#">Section 4.2.3</a>
par	Defined in <a href="#">Section 4.2.1</a>
ref	Defined in <a href="#">Section 4.2.3</a>
seq	Defined in <a href="#">Section 4.2.2</a>
switch	Defined in <a href="#">Section 4.3</a>
text	Defined in <a href="#">Section 4.2.3</a>
textstream	Defined in <a href="#">Section 4.2.3</a>
video	Defined in <a href="#">Section 4.2.3</a>

All of these elements may appear multiple times as children of a "switch" element.

If the "switch" element is used within a "head" element, it can contain the following child:

layout	Defined in <a href="#">Section 3.2</a> Multiple layout elements may occur within the switch element.
--------	---

## 4.4 Test Attributes

This specification defines a list of test attributes that can be added to any synchronization element, and that test system capabilities and settings. Conceptually, these attributes represent boolean tests. When one of the test

attributes specified for an element evaluates to "false", the element carrying this attribute is ignored.

Within the list below, the concept of "user preference" may show up. User preferences are usually set by the playback engine using a preferences dialog box, but this specification does not place any restrictions on how such preferences are communicated from the user to the SMIL player.

The following test attributes are defined in SMIL 1.0:

#### system-bitrate

This attribute specifies the approximate bandwidth, in bits per second available to the system. The measurement of bandwidth is application specific, meaning that applications may use sophisticated measurement of end-to-end connectivity, or a simple static setting controlled by the user. In the latter case, this could for instance be used to make a choice based on the users connection to the network. Typical values for modem users would be 14400, 28800, 56000 bit/s etc. Evaluates to "true" if the available system bitrate is equal to or greater than the given value. Evaluates to "false" if the available system bitrate is less than the given value.

The attribute can assume any integer value greater than 0. If the value exceeds an implementation-defined maximum bandwidth value, the attribute always evaluates to "false".

#### system-captions

This attribute allows authors to distinguish between a redundant text equivalent of the audio portion of the presentation (intended for a audiences such as those with hearing disabilities or those learning to read who want or need this information) and text intended for a wide audience. The attribute can has the value "on" if the user has indicated a desire to see closed-captioning information, and it has the value "off" if the user has indicated that they don't wish to see such information. Evaluates to "true" if the value is "on", and evaluates to "false" if the value is "off".

#### system-language

The attribute value is a comma-separated list of language names as defined in [RFC1766].

Evaluates to "true" if one of the languages indicated by user preferences exactly equals one of the languages given in the value of this parameter, or if one of the languages indicated by

user preferences exactly equals a prefix of one of the languages given in the value of this parameter such that the first tag character following the prefix is "-".

Evaluates to "false" otherwise.

Note: This use of a prefix matching rule does not imply that language tags are assigned to languages in such a way that it is always true that if a user understands a language with a certain tag, then this user will also understand all languages with tags for which this tag is a prefix.

The prefix rule simply allows the use of prefix tags if this is the case.

Implementation note: When making the choice of linguistic preference available to the user, implementors should take into account the fact that users are not familiar with the details of language matching as described above, and should provide appropriate guidance. As an example, users may assume that on selecting "en-gb", they will be served any kind of English document if British English is not available. The user interface for setting user preferences should guide the user to add "en" to get the best matching behavior.

Multiple languages MAY be listed for content that is intended for multiple audiences. For example, a rendition of the "Treaty of Waitangi", presented simultaneously in the original Maori and English versions, would call for:

```
<audio src="foo.rm" system-language="mi, en" />
```

However, just because multiple languages are present within the object on which the system-language test attribute is placed, this does not mean that it is intended for multiple linguistic audiences. An example would be a beginner's language primer, such as "A First Lesson in Latin," which is clearly intended to be used by an English-literate audience. In this case, the system-language test attribute should only include "en".

Authoring note: Authors should realize that if several alternative language objects are enclosed in a "switch", and none of them matches, this may lead to situations such as a video being shown without any audio track. It is thus recommended to include a "catch-all" choice at the end of such a switch which is acceptable in all cases.

#### system-overdub-or-caption

This attribute is a setting which determines if users prefer overdubbing or captioning when the option is available. The attribute can have the values "caption" and "overdub". Evaluates to "true" if the user preference matches this attribute value. Evaluates to "false" if they do not match.

#### system-required

This attribute specifies the name of an extension. Evaluates to "true" if the extension is supported by the implementation, otherwise, this evaluates to "false". In a future version of SMIL, this attribute value will be an XML namespace [NAMESPACES].

#### system-screen-size

Attribute values have the following syntax:

screen-size-val ::= screen-height "x" screen-width

Each of these is a pixel value, and must be an integer value greater than 0. Evaluates to "true" if the SMIL playback engine is capable of displaying a presentation of the given size. Evaluates to "false" if the SMIL playback engine is only capable of displaying a smaller presentation.

#### system-screen-depth

This attribute specifies the depth of the screen color palette in bits required for displaying the element. The value must be greater than 0. Typical values are 1, 8, 24 .... Evaluates to "true" if the SMIL playback engine is capable of displaying images or video with the given color depth. Evaluates to "false" if the SMIL playback engine is only capable of displaying images or video with a smaller color depth.

## Examples

### 1) Choosing between content with different bitrate

In a common scenario, implementations may wish to allow for selection via a "system-bitrate" parameter on elements. The media player evaluates each of the "choices" (elements within the switch) one at a time, looking for an acceptable bitrate given the known characteristics of the link between the media player and media server.

```
...
<par>
  <text .../>
  <switch>
    <par system-bitrate="40000">
      ...
    </par>
    <par system-bitrate="24000">
      ...
    </par>
    <par system-bitrate="10000">
      .....
    </par>
  </switch>
</par>
...
```

### 2) Choosing between audio resources with different bitrate

The elements within the switch may be any combination of elements. For instance, one could merely be specifying an alternate audio track:

```
...
<switch>
  <audio src="joe-audio-better-quality" system-bitrate="16000" />
  <audio src="joe-audio" system-bitrate="8000" />
</switch>
...
```

### 3) Choosing between audio resources in different languages

In the following example, an audio resource is available both in French and in English. Based on the user's preferred language, the player can choose one of these audio resources.

```
...
<switch>
  <audio src="joe-audio-french" system-language="fr" />
  <audio src="joe-audio-english" system-language="en" />
</switch>
...
```

### 4) Choosing between content written for different screens

In the following example, the presentation contains alternative parts designed for screens with different resolutions and bit-depths. Depending on the particular characteristics of the screen, the player can choose one of the alternatives.

```
...
<par>
  <text .../>
  <switch>
    <par system-screen-size="1280X1024" system-screen-depth="16">
      .....
    </par>
    <par system-screen-size="640X480" system-screen-depth="32">
      ...
    </par>
    <par system-screen-size="640X480" system-screen-depth="16">
      ...
    </par>
  </switch>
</par>
...
```

### 5) Distinguishing caption tracks from stock tickers

In the following example, captions are shown only if the user wants captions on.

```
...
<seq>
  <par>
    <audio      src="audio.rm" />
    <video      src="video.rm" />
    <textstream src="stockticker.rtx" />
    <textstream src="closed-caps.rtx" system-captions="on" />
  </par>
</seq>
...
```

### 6) Choosing the language of overdub and caption tracks

In the following example, a French-language movie is available with English, German, and Dutch overdub and caption tracks. The following SMIL segment expresses this, and switches on the alternatives that the user prefers.

```
...
<par>
  <switch>
    <audio src="movie-aud-en.rm" system-language="en"
           system-overdub-or-caption="overdub" />
    <audio src="movie-aud-de.rm" system-language="de"
           system-overdub-or-caption="overdub" />
    <audio src="movie-aud-nl.rm" system-language="nl"
           system-overdub-or-caption="overdub" />
    <!-- French for everyone else -->
    <audio src="movie-aud-fr.rm" />
  </switch>
  <video src="movie-vid.rm" />
  <switch>
    <textstream src="movie-caps-en.rtx" system-language="en"
               system-overdub-or-caption="caption" />
    <textstream src="movie-caps-de.rtx" system-language="de"
               system-overdub-or-caption="caption" />
    <textstream src="movie-caps-nl.rtx" system-language="nl"
               system-overdub-or-caption="caption" />
    <!-- French captions for those that really want them -->
    <textstream src="movie-caps-fr.rtx" system-captions="on" />
  </switch>
</par>
...
```

## 4.5 Hyperlinking Elements

The link elements allows the description of navigational links between objects.

SMIL provides only for in-line link elements. Links are limited to uni-directional single-headed links (i.e. all links have exactly one source and one destination resource). All links in SMIL are actuated by the user.

### Handling of Links in Embedded Documents

Due to its integrating nature, the presentation of a SMIL document may involve other (non-SMIL) applications or plug-ins. For example, a SMIL browser may use an HTML plug-in to display an embedded HTML page. Vice versa, an HTML browser may use a SMIL plug-in to display a SMIL document embedded in an HTML page.

In such presentations, links may be defined by documents at different levels and conflicts may arise. In this case, the link defined by the containing document should take precedence over the link defined by the embedded object. Note that since this might require communication between the browser and the plug-in, SMIL implementations may choose not to comply with this recommendation.

If a link is defined in an embedded SMIL document, traversal of the link affects only the embedded SMIL document.

If a link is defined in a non-SMIL document which is embedded in a SMIL document, link traversal can only affect the presentation of the embedded document and not the presentation of the containing SMIL document. This restriction may be released in future versions of SMIL.

### Addressing

SMIL supports name fragment identifiers and the '#' connector. This means that SMIL supports locators as currently used in HTML (e.g. it uses locators of the form "http://foo.com/some/path#anchor1").



## Linking to SMIL Fragments

A locator that points to a SMIL document may contain a fragment part (e.g. <http://www.w3.org/test.smi#par1>). The fragment part is an id value that identifies one of the elements within the referenced SMIL document. If a link containing a fragment part is followed, the presentation should start as if the user had fast-forwarded the presentation represented by the destination document to the effective begin of the element designated by the fragment.

The following special cases can occur:

1. The element addressed by the link has a "repeat" attribute.
  1. If the value of the "repeat" attribute is N, all N repetitions of the element are played.
  2. If the value of the "repeat" attribute is "indefinite", playback ends according to the rules defined for repeat value "indefinite".
2. The element addressed by the link is contained within another element that contains a "repeat" attribute.
  1. If the value of the "repeat" attribute is N, playback starts at the beginning of the element addressed by the link, followed by N-1 repetitions of the element containing the "repeat" attribute.
  2. If the value of the "repeat" attribute is "indefinite", playback starts at the beginning of the element addressed by the link. Playback ends according to the rules defined for repeat value "indefinite".
3. The element addressed by the link is content of a "switch" element: It is forbidden to link to elements that are the content of "switch" elements.

### 4.5.1 The a Element

The functionality of the "a" element is very similar to the functionality of the "a" element in HTML 4.0 [\[HTML40\]](#). SMIL adds an attribute "show" that controls the temporal behavior of the source when the link is followed. For synchronization purposes, the "a" element is transparent, i.e. it does not influence the synchronization of its child elements. "a" elements may not be nested. An "a" element must have an [href](#) attribute.

#### Attributes

The "a" element can have the following attributes:

- |                      |   |
|----------------------|---|
| id                   | Defined in <a href="#">Section 2</a>  |
| href                 | This attribute contains the URI of the link's destination.<br>The "href" attribute is required for "a" elements.  |
| <a href="#">show</a> | This attribute controls the behavior of the source document containing the link when the link is followed. It can have one of the following values: <ul style="list-style-type: none"> <li>• "replace": The current presentation is paused at its current state and is replaced by the destination resource. If the player offers a history mechanism, the source presentation resumes from the state in which it was paused when the user returns to it.</li> <li>• "new": The presentation of the destination resource starts in a new context, not affecting the source resource.</li> <li>• "pause": The source presentation is paused at its current state, and the destination resource starts in a new context. When the display of the destination resource ends, the source presentation resumes from the state in which it was paused.</li> </ul> |

The default value of "show" is "replace".

title

Defined in [Section 3.3.1](#)

It is strongly recommended that all "a" elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

## Element Content

The "a" element can contain the following children:

animation

Defined in [Section 4.2.3](#)

audio

Defined in [Section 4.2.3](#)

img

Defined in [Section 4.2.3](#)

par

Defined in [Section 4.2.1](#)

ref

Defined in [Section 4.2.3](#)

seq

Defined in [Section 4.2.2](#)

switch

Defined in [Section 4.3](#)

text

Defined in [Section 4.2.3](#)

textstream

Defined in [Section 4.2.3](#)

video

Defined in [Section 4.2.3](#)

## Examples

### *Example 1*

The link starts up the new presentation replacing the presentation that was playing.

```
<a href="http://www.cwi.nl/somewhereelse.smi">
  <video src="rtsp://foo.com/graph.imf" region="l_window" />
</a>
```

In the example, the second line can be replaced by a reference to any valid subtree of an SMIL presentation.

### *Example 2*

The link starts up the new presentation in addition to the presentation that was playing.

```
<a href="http://www.cwi.nl/somewhereelse.smi" show="new">
  <video src="rtsp://foo.com/graph.imf" region="l_window" />
</a>
```

For example, this allows a SMIL player to spawn off an HTML browser.

### *Example 3*

The link starts up the new presentation and pauses the presentation that was playing.

```
<a href="http://www.cwi.nl/somewhereelse.smi" show="pause">
  <video src="rtsp://foo.com/graph.imf" region="l_window"/>
</a>
```

#### Example 4

The following example contains a link from an element in one presentation A to the middle of another presentation B. This would play presentation B starting from the effective begin of the element with id "next".

Presentation A:

```
<a href="http://www.cwi.nl/presentationB#next">
  <video src="rtsp://foo.com/graph.imf"/>
</a>
```

Presentation B (<http://www.cwi.nl/presentation>):

```
...
<seq>
  <video src="rtsp://foo.com/graph.imf"/>
  <par>
    <video src="rtsp://foo.com/timbl.rm" region="l_window"/>
    <video id="next" src="rtsp://foo.com/v1.rm" region="r_window"/>
    ^^^^^^^^^
    <text src="rtsp://foo.com/caption1.html" region="l_2_title"/>
    <text src="rtsp://foo.com/caption2.rtx" region="r_2_title"/>
  </par>
</seq>
...
```

### 4.5.2 The anchor Element

The functionality of the "a" element is restricted in that it only allows associating a link with a complete media object. HTML image maps have demonstrated that it is useful to associate links with spatial subparts of an object. The anchor element realizes similar functionality for SMIL:

1. The anchor element allows associating a link destination to spatial and temporal subparts of a media object, using the "href" attribute (in contrast, the "a" element only allows associating a link with a complete media object).
2. The anchor element allows making a subpart of the media object the destination of a link, using the "id" attribute.
3. The anchor element allows breaking up an object into spatial subparts, using the "coords" attribute.
4. The anchor element allows breaking up an object into temporal subparts, using the "begin" and "end" attributes. The values of the begin and end attributes are relative to the beginning of the media object.

#### Attributes

The anchor element can have the following attributes:

begin

Defined in [Section 4.2.1](#)

coords

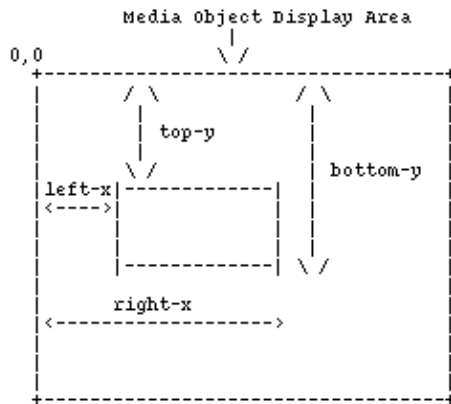
The value of this attribute specifies a rectangle within the display area of a visual media object. Syntax and semantics of this attribute are similar to the coords attribute in HTML image maps, when the link is associated with a rectangular shape. The rectangle is specified by four length values: The first two values specify the coordinates of the upper left corner of the rectangle. The second two values specify the coordinates of the lower right corner of the rectangle. Coordinates are relative to the top left corner of the

visual media object (see Figure 4.5). If a coordinate is specified as a percentage value, it is relative to the total width or height of the media object display area.

An attribute with an erroneous coords value is ignored (right-x smaller or equal to left-x, bottom-y smaller or equal to top-y). If the rectangle defined by the coords attribute exceeds the area covered by the media object, exceeding height and width are clipped at the borders of the media object.

Values of the coords attribute have the following syntax:

coords-value ::= left-x "," top-y "," right-x "," bottom-y



D

Figure 4.5: Semantics of "coords" attribute

end

Defined in [Section 4.2.1](#)

id

Defined in [Section 2](#)

show

Defined in [Section 4.5.1](#)

skip-content

Defined in [Section 3.3.1](#)

title

Defined in [Section 3.3.1](#)

It is strongly recommended that all anchor elements have a "title" attribute with a meaningful description. Authoring tools should ensure that no element can be introduced into a SMIL document without this attribute.

## Examples

### 1) Associating links with spatial subparts

In the following example, the screenspace taken up by a video clip is split into two sections. A different link is associated with each of these sections.

```
<video src="http://www.w3.org/CoolStuff">
  <anchor href="http://www.w3.org/AudioVideo" coords="0%,0%,50%,50%"/>
  <anchor href="http://www.w3.org/Style" coords="50%,50%,100%,100%"/>
</video>
```

### 2) Associating links with temporal subparts

In the following example, the duration of a video clip is split into two subintervals. A different link is associated with each of these subintervals.

```
<video src="http://www.w3.org/CoolStuff">
  <anchor href="http://www.w3.org/AudioVideo" begin="0s" end="5s"/>
  <anchor href="http://www.w3.org/Style" begin="5s" end="10s"/>
</video>
```

### 3) *Jumping to a subpart of an object*

The following example contains a link from an element in one presentation A to the middle of a video object contained in another presentation B. This would play presentation B starting from second 5 in the video (i.e. the presentation would start as if the user had fast-forwarded the whole presentation to the point at which the designated fragment in the "CoolStuff" video begins).

Presentation A:

```
<a href="http://www.cwi.nl/mm/presentationB#tim">
  <video id="graph" src="rtsp://foo.com/graph.imf" region="l_window"/>
</a>
```

Presentation B:

```
<video src="http://www.w3.org/CoolStuff">
  <anchor id="joe" begin="0s" end="5s"/>
  <anchor id="tim" begin="5s" end="10s"/>
</video>
```

### 4) *Combining different uses of links*

The following example shows how the different uses of associated links can be used in combination.

Presentation A:

```
<a href="http://www.cwi.nl/mm/presentationB#tim">
  <video id="graph" src="rtsp://foo.com/graph.imf" region="l_window"/>
</a>
```

Presentation B:

```
<video src="http://www.w3.org/CoolStuff">
  <anchor id="joe" begin="0s" end="5s" coords="0%,0%,50%,50%"
    href="http://www.w3.org/" />
  <anchor id="tim" begin="5s" end="10s" coords="0%,0%,50%,50%"
    href="http://www.w3.org/Tim" />
</video>
```

## 5 SMIL DTD

### 5.1 Relation to XML

A SMIL 1.0 document may optionally contain a document type declaration, which names the document type definition (DTD) in use for the document. For SMIL, the document type declaration should look as follows (the double quotes can be replaced by single quotes):

```
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 1.0//EN"
  "http://www.w3.org/TR/REC-smil/SMIL10.dtd">
```

The XML 1.0 specification provides a way to extend the DTD using the `<!DOCTYPE>` element, for instance to add a new set of entity definitions. Authors must not use this feature with SMIL since many SMIL players will not support it.

The following is illegal in SMIL:

```
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 1.0//EN"
           "http://www.w3.org/TR/REC-smil/SMIL10.dtd" [
<!ENTITY % AcmeCorpSymbols PUBLIC
           "-//Acme Corp//ENTITIES Corporate Symbols//EN"
           "http://www.acme.com/corp_symbols.xml"
>
%AcmeCorpSymbols;
]>
```

## 5.2 DTD

```
<!--
```

This is the XML document type definition (DTD) for SMIL 1.0.

Date: 1998/06/15 08:56:30

Authors:

Jacco van Ossenbruggen <jrvosse@cw.nl>  
Sjoerd Mullender <sjoerd@cw.nl>

Further information about SMIL is available at:

<http://www.w3.org/AudioVideo/>

```
-->
```

```
<!-- Generally useful entities -->
<!ENTITY % id-attr "id ID #IMPLIED">
<!ENTITY % title-attr "title CDATA #IMPLIED">
<!ENTITY % skip-attr "skip-content (true|false) 'true'">
<!ENTITY % desc-attr "
    %title-attr;
    abstract          CDATA    #IMPLIED
    author            CDATA    #IMPLIED
    copyright         CDATA    #IMPLIED
">
```

```
<!--===== SMIL Document =====>
<!--
```

The root element SMIL contains all other elements.

```
-->
<!ELEMENT smil (head?,body?)>
<!ATTLIST smil
    %id-attr;
>
```

```
<!--===== The Document Head =====>
<!ENTITY % layout-section "layout|switch">
```

```
<!ENTITY % head-element "(meta*,((%layout-section;), meta*))?">
```

```
<!ELEMENT head %head-element;>
<!ATTLIST head %id-attr;>
```

```
<!--===== Layout Element =====>
```

```

<!--
    Layout contains the region and root-layout elements defined by
    smil-basic-layout or other elements defined an external layout
    mechanism.
-->
<!ELEMENT layout ANY>
<!ATTLIST layout
    %id-attr;
    type CDATA          "text/smil-basic-layout"
>

<!--===== Region Element =====>
<!ENTITY % viewport-attrs "
    height          CDATA      #IMPLIED
    width           CDATA      #IMPLIED
    background-color CDATA      #IMPLIED
">

<!ELEMENT region EMPTY>
<!ATTLIST region
    %id-attr;
    %title-attr;
    %viewport-attrs;
    left          CDATA      "0"
    top           CDATA      "0"
    z-index       CDATA      "0"
    fit           (hidden|fill|meet|scroll|slice)      "hidden"
    %skip-attr;
>

<!--===== Root-layout Element =====>
<!ELEMENT root-layout EMPTY>
<!ATTLIST root-layout
    %id-attr;
    %title-attr;
    %viewport-attrs;
    %skip-attr;
>

<!--===== Meta Element=====>
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    name          NMTOKEN #REQUIRED
    content CDATA    #REQUIRED
    %skip-attr;
>

<!--===== The Document Body =====>
<!ENTITY % media-object "audio|video|text|img|animation|textstream|ref">
<!ENTITY % schedule "par|seq|(%media-object;)">
<!ENTITY % inline-link "a">
<!ENTITY % assoc-link "anchor">
<!ENTITY % link "%inline-link;">
<!ENTITY % container-content "(%schedule;)|switch|(%link;)">
<!ENTITY % body-content "(%container-content;)">

<!ELEMENT body (%body-content;)*>
<!ATTLIST body %id-attr;>

<!--===== Synchronization Attributes =====>
<!ENTITY % sync-attributes "
    begin CDATA      #IMPLIED
    end   CDATA      #IMPLIED

```

">

<!--===== Switch Parameter Attributes =====>

<!ENTITY % system-attribute "

|                           |                   |          |
|---------------------------|-------------------|----------|
| system-bitrate            | CDATA             | #IMPLIED |
| system-language           | CDATA             | #IMPLIED |
| system-required           | NMTOKEN           | #IMPLIED |
| system-screen-size        | CDATA             | #IMPLIED |
| system-screen-depth       | CDATA             | #IMPLIED |
| system-captions           | (on off)          | #IMPLIED |
| system-overdub-or-caption | (caption overdub) | #IMPLIED |

">

<!--===== Fill Attribute =====>

<!ENTITY % fill-attribute "

|      |                 |          |
|------|-----------------|----------|
| fill | (remove freeze) | 'remove' |
|------|-----------------|----------|

">

<!--===== The Parallel Element =====>

<!ENTITY % par-content "%container-content;">

<!ELEMENT par (%par-content;)\*>

<!ATTLIST par

|                    |                |
|--------------------|----------------|
| %id-attr;          |                |
| %desc-attr;        |                |
| endsync            | CDATA "last"   |
| dur                | CDATA #IMPLIED |
| repeat             | CDATA "1"      |
| region             | IDREF #IMPLIED |
| %sync-attributes;  |                |
| %system-attribute; |                |

>

<!--===== The Sequential Element =====>

<!ENTITY % seq-content "%container-content;">

<!ELEMENT seq (%seq-content;)\*>

<!ATTLIST seq

|                    |                |
|--------------------|----------------|
| %id-attr;          |                |
| %desc-attr;        |                |
| dur                | CDATA #IMPLIED |
| repeat             | CDATA "1"      |
| region             | IDREF #IMPLIED |
| %sync-attributes;  |                |
| %system-attribute; |                |

>

<!--===== The Switch Element =====>

<!-- In the head, a switch may contain only layout elements,  
in the body, only container elements. However, this  
constraint cannot be expressed in the DTD (?), so  
we allow both:

-->

<!ENTITY % switch-content "layout|(%container-content;)">

<!ELEMENT switch (%switch-content;)\*>

<!ATTLIST switch

|              |
|--------------|
| %id-attr;    |
| %title-attr; |

>

<!--===== Media Object Elements =====>

<!-- SMIL only defines the structure. The real media data is  
referenced by the src attribute of the media objects.

-->

<!-- Furthermore, they have the following attributes as defined



in the SMIL specification:

```
-->
<!ENTITY % mo-attributes "
    %id-attr;
    %desc-attr;
    region      IDREF      #IMPLIED
    alt         CDATA      #IMPLIED
    longdesc    CDATA      #IMPLIED
    src         CDATA      #IMPLIED
    type        CDATA      #IMPLIED
    dur         CDATA      #IMPLIED
    repeat      CDATA      '1'
    %fill-attribute;
    %sync-attributes;
    %system-attribute;
">

<!--
Most info is in the attributes, media objects are empty or
contain associated link elements:
-->
<!ENTITY % mo-content "(%assoc-link;)*">
<!ENTITY % clip-attrs "
    clip-begin    CDATA    #IMPLIED
    clip-end      CDATA    #IMPLIED
">

<!ELEMENT ref          %mo-content;>
<!ELEMENT audio        %mo-content;>
<!ELEMENT img          %mo-content;>
<!ELEMENT video        %mo-content;>
<!ELEMENT text         %mo-content;>
<!ELEMENT textstream   %mo-content;>
<!ELEMENT animation    %mo-content;>

<!ATTLIST ref          %mo-attributes; %clip-attrs;>
<!ATTLIST audio        %mo-attributes; %clip-attrs;>
<!ATTLIST video        %mo-attributes; %clip-attrs;>
<!ATTLIST animation    %mo-attributes; %clip-attrs;>
<!ATTLIST textstream   %mo-attributes; %clip-attrs;>
<!ATTLIST text         %mo-attributes;>
<!ATTLIST img          %mo-attributes;>

<!--===== Link Elements =====>

<!ENTITY % smil-link-attributes "
    %id-attr;
    %title-attr;
    href        CDATA      #REQUIRED
    show        (replace|new|pause)  'replace'
">

<!--===== Inline Link Element =====>
<!ELEMENT a (%schedule;|switch)*>
<!ATTLIST a
    %smil-link-attributes;
>

<!--===== Associated Link Element =====>
<!ELEMENT anchor EMPTY>
<!ATTLIST anchor
    %skip-attr;
    %smil-link-attributes;
```

```
%sync-attributes;
coords          CDATA          #IMPLIED
```

&gt;

## References

### [CSS2]

"Cascading Style Sheets, level 2", B. Bos, H. Lie, C. Lilley, I. Jacobs, 12 May 1998.  
Available at <http://www.w3.org/TR/REC-CSS2/>.

### [HTML40]

"HTML 4.0 Specification", D. Raggett, A. Le Hors, I. Jacobs, 24 April 1998.  
Available at <http://www.w3.org/TR/REC-html40/>.

### [ISO/IEC 10646]

ISO (International Organization for Standardization). ISO/IEC 10646-1993 (E). Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane. [Geneva]: International Organization for Standardization, 1993 (plus amendments AM 1 through AM 7).

### [NAMESPACES]

"Namespaces in XML", T. Bray, D. Hollander, A. Layman, 27 March 1998  
W3C working draft. Available at <http://www.w3.org/TR/WD-xml-names>.

### [PICS]

"PICS 1.1 Label Distribution -- Label Syntax and Communication Protocols", 31 October 1996, T. Krauskopf, J. Miller, P. Resnick, W. Trees  
Available at <http://www.w3.org/TR/REC-PICS-labels-961031>

### [RFC1738]

"Uniform Resource Locators", T. Berners-Lee, L. Masinter, and M. McCahill, December 1994.  
Available at <ftp://ftp.isi.edu/in-notes/rfc1738.txt>.

### [RFC1766]

"Tags for the Identification of Languages", H. Alvestrand, March 1995.  
Available at <ftp://ftp.isi.edu/in-notes/rfc1766.txt>.

### [RFC1808]

"Relative Uniform Resource Locators", R. Fielding, June 1995.  
Available at <ftp://ftp.isi.edu/in-notes/rfc1808.txt>.

### [RFC2045]

"Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed and N. Borenstein, November 1996.  
Available at <ftp://ftp.isi.edu/in-notes/rfc2045.txt>. Note that this RFC obsoletes RFC1521, RFC1522, and RFC1590.

### [SMPTE]

"Time and Control Codes for 24, 25 or 30 Frame-Per-Second Motion-Picture Systems - RP 136-1995".  
Society of Motion Picture & Television Engineers.

### [URI]

"Uniform Resource Identifiers (URI): Generic Syntax and Semantics", T. Berners-Lee, R. Fielding, L. Masinter, 4 March 1998.  
Available at <http://www.ics.uci.edu/pub/ietf/uri/draft-fielding-uri-syntax-02.txt>. This is a work in progress that is expected to update [RFC1738] and [RFC1808].

### [XML10]

"Extensible Markup Language (XML) 1.0", T. Bray, J. Paoli, C.M. Sperberg-McQueen, editors, 10 February 1998.  
Available at <http://www.w3.org/TR/REC-xml/>

## Appendix

### Extending SMIL 1.0

(*non-normative*)

In the future, SMIL 1.0 may be extended by another W3C recommendation, or by private extensions.

For these extensions, it is recommended that the following rules are obeyed:

- All elements introduced in extensions must have a "skip-content" attribute (defined in Section 3.3.1) if it should be possible that their content is processed by SMIL 1.0 players.
- Private extensions must be introduced using the syntax of the XML namespace specification.

It is recommended that SMIL 1.0 players are prepared to handle documents that contain extension that obey these two rules.

Extensions should be handled using an XML namespace mechanism, once such a mechanism becomes a W3C recommendation. In the rest of the section, the syntax and semantics for XML namespaces defined in the W3C note [NAMESPACE] will be used for demonstration purposes only.

The following cases can occur:

1. The document contains a namespace declaration for the SMIL 1.0 specification that defines an empty prefix. In this case, non-SMIL 1.0 elements and attributes are only allowed in a document if they are declared using an XML namespace. The document may not contain a document type declaration for SMIL 1.0. If it does, it is invalid.

In the following example, the element "new:a" is a legal extension. The elements "mytags:a" and "b" are syntax errors, since they are not declared using an XML namespace.

```
<?xml:namespace ns="http://www.acme.com/new-smil" prefix="new" ?>
<?xml:namespace ns="http://www.w3.org/TR/PR-smil" ?>
<smil>
  <body>
    <par>
      <new:a>
        ...
      </new:a>
      <mytags:a ... />
      ...
    </mytags:a>
    <b>
      ...
    </b>
    </par>
  </body>
</smil>
```

2. The document contains no document type declaration, it contains a document type declaration for a SMIL version higher than 1.0, or it contains a namespace declaration for a SMIL specification with a version higher than 1.0. For a SMIL 1.0 player to be able to recognize such a namespace declaration, it is recommended that the URI of future SMIL versions starts with <http://www.w3.org/TR/REC-smil>, and is followed by more characters which may for example be a version number.

In this case, a SMIL 1.0 player should assume that it is processing a SMIL document with a version number higher than 1.0.

The following cases can occur:

#### Unknown element

Unknown elements are ignored

An unknown element may contain content that consists of SMIL 1.0 elements. Whether such content is ignored or processed depends on the value of the "skip-content" attribute. If the attribute

is set to "true", or the attribute is absent, the content is not processed. If it is set to "false", the content is processed.

#### Content in Element that was declared "Empty"

A future version of SMIL may allow content in elements that are declared as "empty" in SMIL 1.0. Whether this content is ignored or not depends on the value of the "skip-content" attribute of the formerly empty element. If the attribute is set to "true", the content is not processed. If it is set to "false", the content is processed.

#### Unknown Attribute

Unknown attributes are ignored.

#### Unknown Attribute Value

Attributes with unknown attribute values are ignored.

3. The document contains a document type declaration for SMIL 1.0. In this case, it may not contain any non-SMIL 1.0 elements, even if they are declared using XML namespaces. This is because such extensions would render the document invalid.

## Using SMIL 1.0 as an Extension

When the XML namespace mechanism is used to include SMIL elements and attributes in other XML-based documents, it is recommended to use the following namespace identifier: `http://www.w3.org/TR/REC-smil`