



INTRODUCTION TO STREAMING MEDIA

with RealOne Player

Last Update: 1 October 2002

RealNetworks, Inc.
PO Box 91123
Seattle, WA 98111-9223
U.S.A.

<http://www.real.com>
<http://www.realn networks.com>

©2002 RealNetworks, Inc. All rights reserved.

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of RealNetworks, Inc.

Printed in the United States of America.

Helix, The Helix Logo, RBN, the Real "bubble" (logo), Real Broadcast Network, RealAudio, Real.com, RealJukebox, RealMedia, RealNetworks, RealPlayer, RealOne, RealPresenter, RealSlideshow, RealSystem, RealText, RealVideo, SureStream, and Surreal.FX Design are trademarks or registered trademarks of RealNetworks, Inc.

Other product and corporate names may be trademarks or registered trademarks of their respective companies.

CONTENTS

INTRODUCTION	1
What is Helix?	1
How this Guide Is Organized	1
How to Download This Guide to Your Computer	3
Conventions Used in this Guide.....	4
Additional Documentation Resources	4
Technical Support	5
1 REALONE PLAYER	7
The Three-Pane Environment	7
The Media Playback Pane	8
Media Playback Pane Sizing	9
Media Playback Pane Alone	10
Media Playback and Related Info Panes.....	10
Visualizations for Audio-Only Clips	11
Double-Size and Full-Screen Modes.....	12
The Related Info Pane.....	12
Related Info Pane Sizing.....	12
Media Clips Set the Minimum Height	13
Media Browser Pane Can Override the Width	13
HTML Page Caching.....	13
The Media Browser Pane.....	14
Now Playing List.....	14
Secondary Browsing Windows	14
Using Media Clips to Open HTML Pages	15
Appending HTML URLs to Clip URLs in a Ram File	15
Embedding HTML URLs Into a Clip.....	15
Using SMIL to Coordinate Clips and HTML Pages.....	15
Controlling a Presentation Through HTML Pages	16
Linking One HTML Pane to the Other.....	16
Launching a Clip with an HTML Page Link.....	17
Using Javascript and VBScript Methods	17
Comparing Production Techniques	17
Opening HTML Pages.....	18

	Managing Clips	18
	Getting Started with Production	19
2	MEDIA PRODUCTION	21
	Audio and Video	21
	Editing Audio and Video Files	22
	Encoding Clips	22
	Other Types of Clips	23
	Animation	23
	Images	23
	RealSlideshow Presentations	23
	RealPix Markup	24
	Streaming Text	24
	Bandwidth Strategies	24
	Audience Bandwidth Targets	25
	SureStream RealAudio and RealVideo	25
	Video Dimensions	27
	When to Lower the Streaming Speed	28
	Delivery Options	29
	Helix Universal Server Streaming	29
	Using Helix Universal Server Through an Internet Service Provider	30
	Web Server Downloading	30
3	THE RAM FILE	33
	Launching Clips with a Ram File	33
	Is a Ram File Necessary?	33
	Writing a Ram File	34
	What URLs Do You Use?	35
	Why Does Helix Universal Server Use RTSP?	36
	Putting Comments In a Ram File	36
	Adding Parameters to Your Ram File	37
	Opening an HTML Page as a Clip Plays	37
	Background Color Values	39
	Examples of Opening HTML Pages	40
	Controlling How a Presentation Initially Displays	40
	Examples of Setting a Clip's Initial Display	41
	Tips for Setting the Initial Display	42
	Overriding Title, Author, and Copyright Information	43
	Example of Setting Title, Author, and Copyright Information	43
	Tips for Using Title, Author, and Copyright Parameters	43
	Setting Clip Information	44
	Using Text Escape Characters	45
	Example of Setting Clip Information	46

	Moving Files to a Server	46
	Using FTP to Transfer Clips	47
	Where Does the Ram File Go?	47
4	CLIP-ENCODED URLS	49
	Using Clip-Encoded URLs	49
	Writing an Events File	50
	Specifying URL Events	50
	Adding Clip Information	51
	Merging the Events File with the Clip	52
	Streaming the Clip	53
5	JAVASCRIPT COMMANDS	55
	Declaring Javascript Methods	55
	Playing a Clip	56
	PlayClip() Arguments	56
	URL	56
	clipinfo	56
	context_url	56
	width and height	57
	media_browser_url	57
	Opening a Page in the Media Browser Pane	57
	Setting the Media Background Color	57
6	SMIL HYPERLINKS	59
	What is SMIL?	59
	Writing a SMIL File	59
	How Does the SMIL File Fit In?	60
	SMIL File Basics	60
	SMIL File Sections	61
	Presentation Information	61
	Clip Source Tags	62
	Clip URLs	63
	Creating Relative Links to Other Directories	63
	Writing Absolute Links	64
	Opening HTML Pages with SMIL	64
	How the <area/> Tag Works	64
	Using Standard Link Attributes	65
	Opening an HTML Page Automatically	65
	Controlling the Media Playback State	66
	Selecting the HTML Pane	66
	Opening HTML Pages in the Related Info Pane	66
	Hyperlinking Examples	67

	Opening Several Web Pages During a Presentation.....	67
	Opening Pages on a Mouse Click	68
	Using Advanced Hyperlinking Features	69
7	SMIL SEQUENCES	71
	Playing Clips in Sequence.....	71
	Creating Sequences Without the <seq> Tag	72
	Laying Out a Sequence of Videos.....	72
	Adding Clip or Group Information	73
	Using SMIL Timing Attributes	75
	Setting a Begin Time.....	75
	Using Internal Clip Begin and End Times	76
	Specifying a Duration	76
	Using Advanced Clip and Timing Features	76
8	SMIL LAYOUT	79
	Setting the Media Pane Size	79
	Considerations For Setting a Root-Layout Size	79
	Defining the Root-Layout	80
	Making Room for the Related Info Pane	81
	Creating Playback Regions	81
	Adding <region/> Tags	82
	Defining Region Sizes and Positions	83
	Layout Example 1: Region Width and Height.....	84
	Layout Example 2: Four Region Offsets	84
	Layout Example 3: Region Sizes and Two Offsets	85
	Layout Example 4: Two Offsets	86
	Layout Example 5: Single Offsets for Two Regions.....	86
	Layout Example 6: Overlapping Regions	87
	Tips for Defining Region Sizes and Offsets	88
	Adding Background Colors	88
	Setting When Background Colors Appear.....	89
	Transparency in Regions and Clips	89
	Positioning Clips in Regions.....	90
	Creating Registration Points	90
	Avoiding Problems When Defining Registration Points	91
	Using Common Registration Point Values in Clip Source Tags	91
	Defining How Clips Fit Regions	92
	Using fit Attribute Values	93
	Tips for Defining the fit Attribute	94
	Assigning Clips to Regions	95
	Playing Clips in Parallel.....	95
	Ending a Parallel Group on a Specific Clip	95

Setting Clip Fills	96
Using Images in Parallel Groups	97
Tips for Creating Parallel Groups.....	97
Defining Groups Within Groups.....	98
Layout Examples	100
Centering a Video on a Background.....	100
Displaying a Letterbox Clip	100
Playing Three Clips Side-by-Side	101
Using Advanced Layout Features	102
A QUICK ANSWERS	103
Playing Media with RealOne Player.....	103
Creating Streaming Clips	104
Getting Production Tools.....	105
Using SureStream	107
Writing SMIL Files	107
Streaming Clips	109
Broadcasting.....	111
Getting Technical Support	113
B SMIL SYNTAX	115
SMIL Extension and File Names.....	115
Tags, Attributes, and Values.....	115
End Tags for Tag Pairs	116
Closing Slash for Single Tags	116
Tag and Attribute Case Sensitivity	116
Tag ID Values.....	116
The SMIL Tag.....	117
SMIL 2.0 Namespace.....	117
RealNetworks Extension Namespace	118
The Header Section	118
The Body Section.....	119
Indentation and Line Returns	119
Comments	120
Summary of SMIL Syntax	120
C SMIL TAG SUMMARY	123
<smil>...</smil>	123
Header Tags.....	123
<meta/>	124
<layout>...</layout>.....	124
<root-layout/>	124
<region/>.....	125

Clip Source Tags	126
<seq>...</seq>	127
<par>...</par>	128
<area/>	128
<i>D</i> RAM FILE SUMMARY	131
Parameter Syntax.....	131
Parameters and Values.....	131
GLOSSARY	133
INDEX	139

INTRODUCTION

RealOne™ Player is the successor to RealPlayer® and RealJukebox®, the world's most popular applications for playing streaming or downloaded media. By combining streaming media, digital downloads, and Internet browsing, RealOne Player provides an all-in-one consumer application for media distribution. This guide introduces you to the production techniques that you can use to create compelling streaming media presentations in RealOne Player.

Tip: To experience the many possibilities of streaming media, download RealOne Player from <http://www.real.com>, and then visit <http://realguide.real.com>. Refer to the RealOne Player **Help** menu for tips about using RealOne Player.

What is Helix?

Helix™ from RealNetworks is a universal digital media delivery platform. With industry-leading performance, integrated content distribution, advertising, user authentication, Web services support, and native delivery of RealMedia, Windows Media, QuickTime, and MPEG-4, Helix from RealNetworks is a robust digital media foundation that meets the needs of enterprises and networking service providers.

How this Guide Is Organized

Chapter 1: RealOne Player

This chapter guides you through RealOne Player's three-pane environment, and introduces you to the various production techniques that you can use to coordinate streaming media with HTML pages.

Chapter 2: Media Production

This chapter outlines media clip production, explaining the types of clips that you can stream, and pointing out important issues to consider when producing streaming media.

Chapter 3: The Ram File

Read this chapter to learn how to write a Ram file, the file that launches RealOne Player and identifies a clip to stream. With a Ram file, you can also specify HTML pages that play along with a media clip.

Chapter 4: Clip-Encoded URLs

This chapter gives you instructions on an alternate way to open HTML pages as a RealAudio or RealVideo clip plays: encoding the page URLs directly into the clip.

Chapter 5: Javascript Commands

Refer to this chapter to learn how to use some of the Javascript functions available within the RealOne Player environment.

Chapter 6: SMIL Hyperlinks

This chapter introduces you to SMIL, a markup language designed specifically for coordinating media clips. It explains how you can use SMIL's timing features to open HTML pages as a clip plays.

Chapter 7: SMIL Sequences

Once you grasp the basics of SMIL as described in Chapter 6, read this chapter to learn how to use SMIL to create a sequence of streaming clips.

Chapter 8: SMIL Layout

Building on the information in Chapter 6 and Chapter 7, this chapter explains how to use SMIL to play multiple clips at the same time.

Appendix A: Quick Answers

Refer to this appendix for quick answers to common questions about producing streaming media.

Appendix B: SMIL Syntax

This appendix provides a reference for SMIL syntax rules. It's important to understand these rules to avoid errors when writing SMIL markup.

Appendix C: SMIL Tag Summary

Once you understand SMIL, use this appendix as a reference for SMIL tag and attribute values.

Appendix D: Ram File Summary

This appendix provides a quick reference for the Ram file parameters that Chapter 3 explains in detail.

How to Download This Guide to Your Computer

RealNetworks makes this guide available in the following formats for download to your computer:

- The HTML+Javascript version is available as a single, zipped archive that includes samples that you can play in RealOne Player. You can read this version with Netscape Navigator or Microsoft Internet Explorer.

Note: If you browse the HTML+Javascript version of this manual with Netscape Navigator 6, you may not be able to play the linked sample files. If this occurs, you can open the sample files directly from the samples folder. This problem affects only local, relative URLs to clips played in RealOne Player. It does not affect streamed presentations in which clips are placed on Web servers or Helix Universal Server, and the viewer launches the presentation from a Web page rendered in Navigator 6.

- The HTML Help version is available as a single .chm file for Windows 98 and later operating systems. It is identical to the HTML+Javascript version, except that it does not contain any sample files. The HTML Help version is smaller in size than the HTML+Javascript version, and it includes a search function.
- An Adobe Acrobat (PDF) version includes page numbers in cross-references, making it more useful than the HTML versions when printed. You can download the free Acrobat viewer from Adobe's Web site at **<http://www.adobe.com/products/acrobat/readstep.html>**.

All of the online versions of this guide are available for individual download from RealNetworks' Technical Support Web site at:

<http://service.real.com/help/library/encoders.html>

Conventions Used in this Guide

The following table explains the typographical conventions used in this guide.

Notational Conventions	
Convention	Meaning
emphasis	Bold text is used for in-line headings, user-interface elements, URLs, and e-mail addresses.
<i>terminology</i>	Italic text is used for technical terms being introduced, and to lend emphasis to generic English words or phrases.
syntax	This font is used for fragments or complete lines of programming syntax (markup).
syntax emphasis	Bold syntax character formatting is used for program names, and to emphasize specific syntax elements.
<i>variables</i>	Italic syntax character formatting denotes variables within fragments or complete lines of syntax.
[options]	Square brackets indicate values that you may or may not need to use. As a rule, when you use these optional values, you do not include the brackets themselves.
choice 1 choice 2	Vertical lines, or “pipes,” separate values that you can choose between.
...	Ellipses indicate nonessential information omitted from examples.

Additional Documentation Resources

In addition to this introductory guide, you may need the following resources, which are available for download at <http://service.real.com/help/library/encoders.html>:

- *Helix Producer User’s Guide*

This user’s guide gives you the step-by-step instructions for running Helix Producer™, which turns audio and video files into streaming RealAudio® and RealVideo® clips. An online version of this guide is available through the Helix Producer **Help** menu.

- *RealNetworks Production Guide*

This guide is the main reference manual for streaming media production. It expands on most of the topics presented in this introductory guide.

Refer to it for instructions and tips on media production, as well as for complete information about using SMIL.

- *RealOne Player Scripting Guide*

If you are a Web programmer, refer to this guide for instructions about using Javascript or VBScript with RealOne Player. Using these scripting languages, you can customize RealOne Player to turn it into your own Internet jukebox, for example.

Technical Support

To reach RealNetworks' Technical Support, please fill out the form at:

- **<http://forms.real.com/service/techsupport/contact.html>**

The information you provide in this form will help Technical Support personnel respond promptly.

REALONE PLAYER

Combining streaming media playback with digital downloads, RealOne Player integrates the features found in previous versions of RealPlayer and RealJukebox, adding Web browsing as well. This chapter describes RealOne Player's three-pane environment, and introduces you to the powerful production techniques that you can use to create compelling streaming media presentations.

The Three-Pane Environment

RealOne Player integrates streaming media with HTML pages simply and effectively. Because previous versions of RealPlayer did not natively display HTML pages, linked pages opened in the viewer's default Web browser, which split the presentation between separate applications. RealOne Player closes this divide, benefitting both the viewer, who does not have to switch between applications to watch an integrated presentation, and the presentation author, who can more easily coordinate streaming media with Web pages.

As with past RealPlayers, you can still embed streaming media in any Web page that viewers display in their favorite Web browsers. Although embedding is a widely used means of integrating streaming media with HTML content, the required embedding markup can be cumbersome. With RealOne Player, you can keep your streaming media and HTML pages separate, coordinating the two with simple production techniques. This reduces the work required to stream media and display HTML pages simultaneously.

The following figure illustrates the three-pane environment of RealOne Player, which is based on the metaphor of "play/more/explore." Here, the Media Playback pane plays streamed or downloaded clips. The Related Info pane gives the viewer more information about the presentation. And the detachable Media Browser pane lets the viewer explore the World Wide Web. This design gives you one pane for playing media, one pane for displaying

small HTML pages related to the media, and one pane for showing large Web pages, such as your home page.

RealOne Player Three-Pane Environment with a Secondary Browsing Window



The Media Playback Pane

The media playback pane hosts media clips and includes buttons for play, pause, rewind, volume control, and so on. Any streaming or downloaded media playable in RealOne Player can display in this pane. This includes the core clip types and markup languages:

- RealAudio[®] or MP3 for audio
- RealVideo[®], MPEG-1, or MPEG-4 for video
- RealText[®] for timed text

- RealPix™ for still-image slideshows
- Macromedia Flash for animation
- SMIL for creating an integrated presentation from multiple clips

In addition, RealOne Player can play many other media types, including MPEG audio and video. Chapter 2 introduces you to the media production concepts and practices that you use to create streaming media clips that play in the media playback pane.

Media Playback Pane Sizing

The media playback pane automatically scales to the size of the playing media. If no HTML page displays in the related info pane as media plays, the media playback pane appears centered above the media browser pane as shown in the following figure. The media browser pane's resize handle allows the viewer to adjust the relative heights of the top and bottom halves of the three-pane environment.

Media Playback Pane Centered Above the Media Browser Pane



Tip: As explained in “Making Room for the Related Info Pane” on page 81, you can use SMIL to display the media playback pane at the left side of the RealOne Player window instead of in the center.

Media Playback Pane Alone

If the viewer has detached or closed the media browser pane, the media playback pane encloses the playing media, as illustrated in the next figure. This gives the viewer access to media in a smaller pane that includes just the necessary controls for adjusting media playback.

Media Playback Pane Without the Media Browser Pane



Media Playback and Related Info Panes

If a media presentation opens an HTML page in the related info pane, the media playback pane automatically expands to display both the media and the HTML page, as shown in the next figure.

Media Playback Pane With the Related Info Pane



Visualizations for Audio-Only Clips

When playing audio-only clips, the viewer can display in the media playback pane a *visualization*, such as an audio analyzer consisting of bars that rise and fall in response to the strength of various audio frequencies.

A Visualization in the Media Playback Pane



Double-Size and Full-Screen Modes

Content authors and viewers can also play media at double-size or full-screen. In full-screen mode, the media playback pane expands to fill the entire computer screen. In this case, no HTML pages in the related info or media browser panes display until the presentation ends, or the viewer exits full-screen mode.

The Related Info Pane

The related info pane, which is also called the “context pane,” appears to the right of the media playback pane. It’s designed to display small HTML pages that supplement media clips. These pages might contain album cover art, copyright information, advertisements, and so on. Although using the related info pane is not required, displaying supplemental HTML pages in this pane greatly enhances the viewing experience. The related info pane can display any HTML page content supported by Microsoft Internet Explorer version 4 or later.

Because the media playback and related info panes are separate, you can easily open multiple HTML pages as a presentation plays, displaying each page at a specific point in the media timeline. You can thereby update the related info pane simply by opening a new HTML page. In contrast, when you embed media in a Web page, updating the page as the media plays can require complicated scripting. RealOne Player thereby lets you focus on your media, and display any number of supplemental HTML pages by using simple production techniques.

Note: Because no divider marks the boundary between the media playback and related info panes, it’s easy to blend the panes by setting the same background colors. For the related info pane, you set the background color in the HTML page. Later sections in this guide explain how to set the media playback pane’s background color through various methods.

Related Info Pane Sizing

The RealOne Player production techniques described in this guide let you set the size of the related info pane. If you do not specify a size, the pane uses a default width of 330 pixels, and a height the same as the media playing in the

media playback pane. If the page content is too large for the specified size, the pane displays scroll bars the same as a standard browser window.

The related info pane's size is fixed for the presentation's duration. As a clip or SMIL presentation plays, the first URL that opens in the related info pane sets the pane size. If a subsequent URL opens in the related info pane while the same clip or presentation plays, any sizing information in that URL is ignored. You can specify a new related info pane size, though, when starting a new clip or SMIL presentation.

Media Clips Set the Minimum Height

You can set the related info pane to a height greater than the media, but not smaller. If your media is 300 pixels high, for example, your related info pane will be 300 pixels high even if you specify a shorter height, such as 200 pixels. However, you can create a related info pane that is taller than your media, such as 400 pixels. In this case, RealOne Player centers the media playback pane vertically alongside the related info pane.

Media Browser Pane Can Override the Width

When the bottom media browser pane is attached to the top two panes, it may increase the width of the related info pane. Suppose that you play a media clip that is 200 pixels wide, and you specify a related info pane width of 300 pixels. If the media browser pane is not attached, the width of the top two panes is 500 pixels. If a 600-pixel-wide media browser pane is attached, though, RealOne Player adds 100 pixels to the related info pane width to increase the overall width of the top panes to 600 pixels.

HTML Page Caching

RealOne Player caches the HTML pages that display in the related info pane for the duration of a presentation. It deletes this cache when a new clip plays. RealOne Player does not normally cache media clips that play in the media playback pane. However, when you use SMIL, you can make RealOne Player cache small clips, such as images, that display in the media playback pane.

For More Information: See the clip source tag chapter in *RealNetworks Production Guide* for more information about RealOne Player's CHTTP caching protocol for small media clips.

The Media Browser Pane

The media browser pane can attach to, or detach from, the media playback and related info panes. When attached, it appears below the two other panes. Detached, it appears as a stand-alone window that the viewer can resize and close independently of the media playback and related info panes. Sending an HTML page URL to a closed media browser pane reopens the pane, however.

Through the media browser pane, RealOne Player users can surf the Web, play CDs, access their personal media libraries, transfer clips to portable players, and so on. Presentation authors can also use this pane to display Web pages associated with a streaming presentation. The pane can display any content supported in Microsoft Internet Explorer version 4 or later, including Javascript. You might use this pane to display your home page after a media presentation plays, for example.

Now Playing List

In the left side of the media browser pane, viewers can display a clickable “Now Playing” list. When the viewer plays a streaming media clip or presentation, the clip or presentation title displays in this list. Additionally, the viewer can build a clip list by dragging media links from an HTML page displayed in the related info or media browser pane.

RealOne Player ‘Now Playing’ List



Secondary Browsing Windows

Like most Web browsers, RealOne Player can display any number of additional browsing windows, which are independent of the three-pane environment. You can display Web pages associated with your presentation in secondary browsing windows, for example. Displaying full Web pages in the media browser pane is preferable in most cases, though, because many viewers are likely to have that pane already attached to the media playback and related

info panes. Additionally, only the media browser pane includes the “Now Playing” list.

Using Media Clips to Open HTML Pages

You can use three different techniques to open URLs in an HTML pane as a media clip plays. These techniques allow you to create “media-driven” presentations, in which supplemental information displays in the HTML panes at a specific point in the media timeline, or in response to viewer interaction with clips. You can use these techniques to carry out tasks such as the following:

- Display advertisements.
- Show purchasing information while a clip or a series of clips plays.
- Highlight video topics with informational bullet points.
- Open up your home page automatically when a presentation ends.

Appending HTML URLs to Clip URLs in a Ram File

You typically launch media clips that play in RealOne Player with a Ram file, which uses the extension `.ram`. As Chapter 3 explains, you can include in the Ram file the URLs for HTML pages that open in the related info pane or media browser pane. This Ram file method is easy to use, and is well-suited for simple presentations, such as a single video clip that displays an HTML page as it plays.

Embedding HTML URLs Into a Clip

When you create a RealVideo or RealAudio clip with Helix Producer, you can write an *events file* that defines one or more URLs that open in a RealOne Player HTML pane at certain points as the clip plays. You then use a utility that embeds the events into the clip. Whenever you stream the clip, the encoded URLs open automatically. Chapter 4 provides more information about this production technique.

Using SMIL to Coordinate Clips and HTML Pages

To lay out and synchronize multiple media clips, you use Synchronized Multimedia Integration Language (SMIL). A SMIL presentation always plays

in the media playback pane, but it can also open HTML pages in the other panes. Using SMIL gives you more control over HTML display than using a Ram file, or encoding URLs directly into clips. Chapter 6 explains the basics of opening HTML pages with SMIL. Later chapters provide more information about using SMIL to coordinate multiple clips.

Tip: This guide covers only the basics of SMIL, which gives you full control over your media presentations. The section “What advanced SMIL features can I use?” on page 108 lists some of the more complex SMIL features that you can learn about in *RealNetworks Production Guide*.

Controlling a Presentation Through HTML Pages

Through HTML pages displaying in the related info pane or media browser pane, you can control the media displaying in the media playback pane, as well as open new HTML pages. These production techniques, which you can mix with the media-based techniques described previously, allow you to create “user-driven” presentations, in which clips and HTML pages display according to viewer action within the HTML panes.

Linking One HTML Pane to the Other

The most basic way to link one HTML pane to another is through a simple hypertext link in the form `<a href>`. You can open a new HTML page in the media browser pane through a hypertext link in the related info pane by adding a `target="_rpbrowser"` attribute to the `<a href>` tag:

```
<a href="URL" target="_rpbrowser">
```

Any other target name will open the HTML page in a secondary window that is detached from the basic three-pane environment.

You should not attempt to open an HTML page in the related info pane with a simple link in the media browser pane, however, because the related info pane URL requires sizing information that you cannot pass in the link. However, the Javascript/VBScript methods described below let you pass this information.

Launching a Clip with an HTML Page Link

If you link to a Ram file with a simple `<a href>` link as described in Chapter 3, the clip or SMIL presentation listed in the Ram file automatically plays in the media playback pane. You do not need to use any additional window targeting attributes. To avoid a file download dialog, though, you can use the Javascript or VBScript methods to play clips when the viewer clicks links.

Using Javascript and VBScript Methods

RealOne Player supports several methods that work with both Javascript and VBScript. Used in HTML pages displaying in the related info pane or media browser pane, these methods give you more control than standard `<a href>` links. They are intended for HTML pages displaying in the RealOne Player environment, however, and not for HTML pages rendered by other browsers.

The Javascript/VBScript methods are well-suited for creating Internet-based audio and video jukeboxes, for example. Using these methods, you can create interactive presentations that add clips to the “Now Playing” list, for example, or play clips based on viewer interaction with forms or elements displayed in the related info or media browser pane.

Chapter 5 explains how to use the basic Javascript methods for creating a link in the related info or media browser pane that plays a clip or opens an HTML page. For information about all of the Javascript and VBScript methods, see *RealOne Player Scripting Guide*. Using these methods, you can populate the “Now Playing” list, preload URLs for later display, and more. The *RealOne Player Scripting Guide* also explains how to use Javascript and VBScript commands when embedding streaming media in a Web page.

Comparing Production Techniques

The following sections compare the features of the four production methods discussed in this guide:

1. Adding parameters to a Ram file (Chapter 3).
2. Encoding information in a clip with an events file (Chapter 4).
3. Using the RealOne Player Javascript methods (Chapter 5).
4. Writing a SMIL 2.0 file (Chapter 6 through Chapter 8).

Opening HTML Pages

The following table compares the production methods for opening HTML pages in the related info and media browser panes as clips play.

Comparison of Production Methods for Opening HTML Pages

HTML Page Feature	Ram File	Events File	Javascript	SMIL 2.0
Open a page in the related info pane?	yes (page 38)	yes (page 50)	yes (page 56)	yes (page 66)
Open a related info page at any point during clip playback?	yes (page 38)	yes (page 50)	yes*	yes (page 67)
Display multiple related info pages for a single clip?	no	yes (page 50)	yes*	yes (page 67)
Set the related info pane size?	yes (page 38)	yes (page 50)	yes (page 57)	yes (page 66)
Open a page in the media browser pane?	yes (page 38)	yes (page 50)	yes (page 57)	yes (page 66)
Open a page in the media browser at any point during clip playback?	no	yes (page 50)	yes*	yes (page 67)
Display multiple media browser pages for a single clip?	no	yes (page 50)	yes*	yes (page 67)
Easily change the HTML page URLs?	yes	no	yes	yes
HTML URLs open in viewer's default browser with RealPlayer 8 and earlier?	no	yes	no	no

* The Javascript methods themselves allow you to open two HTML pages, one in the media browser pane and one in the related info pane, when a clip starts. If you are handy with Javascript, though, you can use these methods within a larger scripting context to open multiple pages for each clip, or open HTML URLs at different points in the presentation, by calling the methods at different times. For more information, see *RealOne Player Scripting Guide*.

Managing Clips

The next table compares the production techniques for managing media clips. Note that the different techniques are not mutually exclusive. For example, only a Ram file can open a presentation in double-size mode. However, you

can use this feature with SMIL presentations, or when playing a clip that also has an encoded events file.

Comparison of Production Methods for Handling Media Clips

Media Clip Feature	Ram File	Events File	Javascript	SMIL 2.0
Play only a portion of a clip?	yes (page 41)	no	no	yes (page 76)
Play clips in sequence?	yes (page 35)	no	yes *	yes (page 71)
Introduce each clip with a special effect?	no	no	no	yes (page 76)
Play multiple clips at the same time?	no	no	no	yes (page 95)
Provide basic title, author, and copyright information for each clip?	yes (page 43)	yes (page 52)	no	yes (page 73)
Provide extended clip information, such as genre and album name?	yes (page 44)	yes (page 51)	yes (page 56)	no
Set the media playback pane's background color?	yes (page 39)	no	yes (page 57)	yes (page 88)
Open a presentation in double-size, full-screen, toolbar, or theater mode?	yes (page 40)	no	no	no
Use with clips other than RealAudio and RealVideo?	yes	no	yes	yes

* See *RealOne Player Scripting Guide* for details about the `AddToNowPlaying` method, which allows you to add clips to the “Now Playing” list.

Getting Started with Production

RealOne Player's three-pane environment allows you to develop many different types of presentations that play media alone, or combine streaming media with HTML pages. The many production techniques that RealOne Player supports let anyone, beginner to expert, create simple to highly complex presentations. Here's how to get started:

1. If you've never produced a streaming media clip before, start with “Chapter 2: Media Production” beginning on page 21. That chapter explains the types of media that you can produce, and introduces you to important streaming concepts.

2. If you want to create a simple presentation, “Chapter 3: The Ram File” beginning on page 33 may provide all the information you need. Using a Ram file, you can stream clips and open HTML pages in the related info and media browser panes. “Chapter 4: Clip-Encoded URLs” beginning on page 49 explains how to do most of the same tasks by encoding URLs directly into clips, which some authors prefer.
3. “Chapter 5: Javascript Commands” beginning on page 55 introduces you to the basic RealOne Player Javascript methods. Using Javascript is optional, but if you are familiar with this scripting language, you’ll find powerful tools in the RealOne Player methods.
4. Once you grasp the basics of media production, you can use SMIL to spice up your presentations. SMIL lets you display multiple clips, use timing commands, and add special effects to presentations. “Chapter 6: SMIL Hyperlinks” beginning on page 59 introduces you to SMIL, and explains how to link to an HTML page from SMIL. Chapter 7 and Chapter 8 provide details about using SMIL in complex media presentations.
5. You can also explore the sample files included with this guide. To view all of the sample files, download the zipped HTML version of this guide as described in “How to Download This Guide to Your Computer” on page 3, and choose **Sample Files** from the pull-down menu.

MEDIA PRODUCTION

If you're new to media streaming, this chapter provides background on media production, and introduces you to important streaming concepts. Although it doesn't cover the steps required to encode streaming clips, it tells you where to find the necessary tools and documentation to do so.

For More Information: See also Appendix A, which provides quick answers to common questions about media production and streaming.

Audio and Video

Audio and video are by far the most popular forms of streaming media on the Internet. Most audio and video clips that play in RealOne Player are encoded as RealAudio and RealVideo, although RealOne Player can also play other formats, such as MPEG-1 and MPEG-4 video, as well as MP3 audio.

To create streaming clips, you start with a digitized audio or video file in a standard, uncompressed format. On Windows, WAV (.wav) and AVI (.avi) are the most popular audio and video formats, respectively. On the Macintosh, QuickTime (.mov) and AIFF (.aiff) are commonly used. Unix users often start with MPEG (.mpg, .mpeg).

Tip: If RealOne Player can open a clip, you typically can stream that type of clip with Helix Universal Server. Only compressed clips stream well, though. Uncompressed AVI is not a good streaming format, for example, because it requires a lot of bandwidth for even a small clip.

Note: Although RealOne Player can play proprietary formats used by other media players, such as Windows Media and QuickTime, it does not support the use of a Ram file or SMIL

with these formats. When streaming one of these formats to RealOne Player, you must author presentations using the markup conventions supported by Windows Media Player or QuickTime Player, respectively.

Editing Audio and Video Files

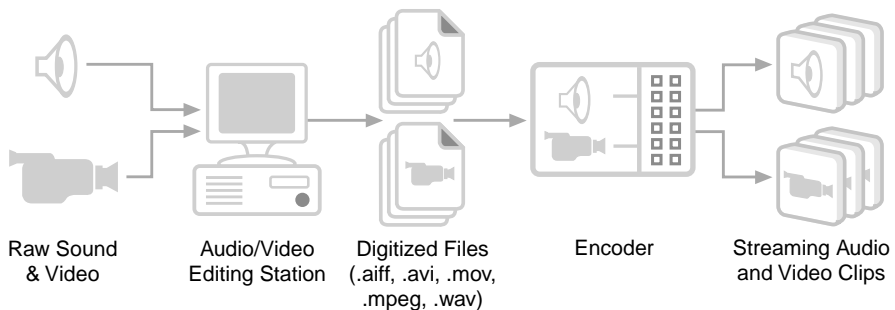
You can use the audio or video editing program of your choice to edit your digitized audio or video file. Using this program, you can set the file length, cropping out any unnecessary parts, for example. Although RealNetworks encoding tools provide some editing functions such as cropping, they do not provide all of the advanced features found in audio and video editing programs, such as tools for sharpening the visual appearance of a video.

Tip: The quality of a streaming audio or video clip starts with the source file. The more you know about audio and video editing, the better you'll be able to produce a great streaming clip. For some pointers about preparing files for encoding, see the audio chapter and video chapter of the *RealNetworks Production Guide*.

Encoding Clips

Some editing programs can export digitized audio and video directly to your streaming formats, such as RealVideo or MPEG-4. If your editing program cannot export clips, or you don't want to use this feature, you can use an encoding tool to encode clips from files in standard formats. Helix Producer Basic is a free tool for encoding RealAudio and RealVideo clips. Helix Producer Plus is an enhanced version that offers more encoding features.

Encoders Create Streaming Clips



For More Information: You can get more information about Helix Producer, as well as download this tool from <http://www.realnworks.com/products/producer/index.html>.

Other Types of Clips

Although audio and video command the biggest share of streaming media on the Web, they are not the only types of clips that you can stream. RealOne Player can play many different types of clips, giving you a lot of ways to combine diverse media into a single presentation. The following sections explain the types of clips in addition to audio and video that can play in the RealOne Player media playback pane.

Animation

With Macromedia Flash animation, you can build anything from streaming cartoons to e-commerce applications. Using version 5 of the Flash application, you can export an animation directly for streaming to RealOne Player, complete with a RealAudio soundtrack. A streaming Flash clip uses the file extension .swf. Learn more about Flash from Macromedia's Web site at:

<http://www.macromedia.com/software/flash/>

Tip: A Flash chapter in *RealNetworks Production Guide* explains how to optimize a Flash clip for streaming, as well as how to encode URLs into the clip.

Images

Still images in the GIF, JPEG, or PNG format can display in the media playback pane, as well as in the media browser and related info panes. Using SMIL, you can create streaming presentations that include images along with audio or video clips. You can even turn the images into interactive buttons. Chapter 8 explains the basics of how to play multiple clips in the media playback pane.

RealSlideshow Presentations

When you want to create a streaming slideshow, the easiest solution is to use RealSlideshow™ or RealSlideshow Plus. These tools have drag-and-drop

interfaces that let you quickly build your slideshow, which can include text captions, audio narrations, and background music. Get RealSlideshow at:

<http://www.realnworks.com/products/index.html>

RealPix Markup

Streaming slideshows are based on the RealPix markup language. Instead of using RealSlideshow, you can write your own markup to assemble images into a RealPix presentation that has eye-catching special effects such as dissolves and zooms. A RealPix markup file uses the file extension .rp. Learn the RealPix markup language from *RealNetworks Production Guide*, available for download from the following Web page:

<http://service.real.com/help/library/encoders.html>

Streaming Text

To create streaming, timed text, you can use RealText, a simple markup language that lets you subtitle videos, for example, or create hypertext links within the media playback pane. A RealText markup file uses the file extension .rt. The RealText chapter of *RealNetworks Production Guide* explains how to write this markup.

Bandwidth Strategies

Any computer connected to a network has a connection bandwidth, which is a maximum speed at which it can receive data. Web users with 28.8 Kbps modems, for example, can view only those presentations that stream less than 28.8 Kb of data per second. Presentations that stream more data than that per second may stall because the data cannot get over the modems fast enough to keep the clips flowing. These presentations will not cause problems for users with faster connections, though.

Successfully targeting your audience's connection bandwidth is crucial for producing streaming media. Viewers don't like to wait more than a few seconds for playback to begin after they click a link. And if your clips sputter because they use too much bandwidth, viewers are not likely to stay tuned. Developing a bandwidth strategy helps ensure that clips play back quickly and don't stall. You can also employ methods for delivering good clips to users with slow connections, and great clips to those with fast connections.

Audience Bandwidth Targets

Your streaming presentations should never consume all of your audience's connection bandwidth. They must always leave bandwidth for network overhead, error correction, resending lost data, and so on. Otherwise, they may frequently pause while waiting for more data to arrive. The following table recommends maximum streaming speeds for common network connections. To reach 28.8 Kbps modems, for example, a presentation should stream no more than 20 Kb of data per second.

Maximum Streaming Rates

Target Audience	Maximum Streaming Rate
14.4 Kbps modem	10 Kbps
28.8 Kbps modem	20 Kbps
56 Kbps modem	34 Kbps
64 Kbps ISDN	45 Kbps
112 Kbps dual ISDN	80 Kbps
Corporate LAN	150 Kbps
256 Kbps DSL/cable modem	225 Kbps
384 Kbps DSL/cable modem	350 Kbps
512 Kbps DSL/cable modem	450 Kbps
786 Kbps DSL/cable modem	700 Kbps

For any other connection speed, calculate the maximum streaming speed as:

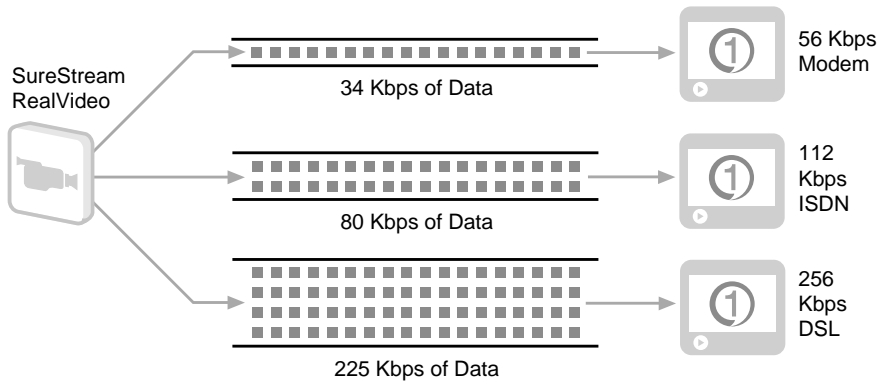
- Approximately 75 percent of the connection bandwidth for analog connections such as dial-up modems.
- Or–
- Approximately 90 percent of the connection bandwidth for high-speed digital connections such as DSL or cable modems.

SureStream RealAudio and RealVideo

Helix Producer encodes your RealAudio or RealVideo clip for the proper bandwidth (or bandwidths) automatically, based on the audiences you choose. Using SureStream technology, you can encode a single clip for multiple bandwidths. For example, you can encode a single RealAudio music clip for 56 Kbps modems, 112 Kbps dual ISDN, 256 Kbps DSL, and so on. The clip's

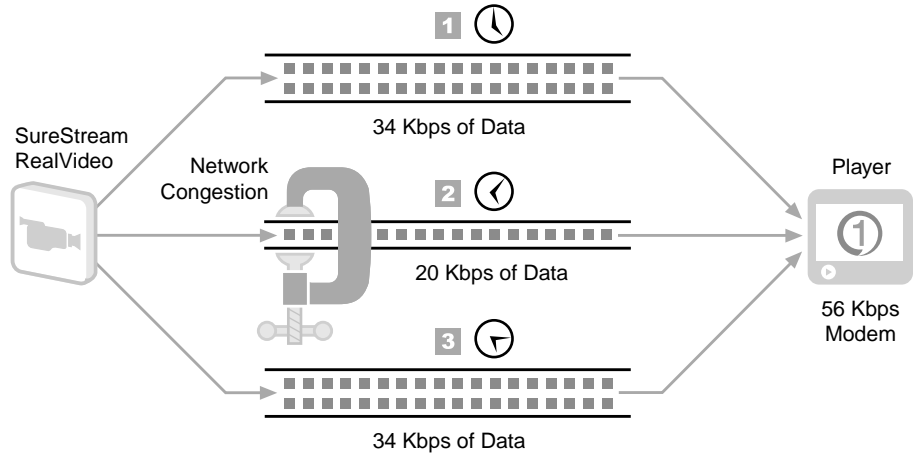
playback quality improves with each faster speed. When a viewer clicks a link to a SureStream clip, RealOne Player and Helix Universal Server determine which stream to use based on the available bandwidth.

SureStream Clip Encoded for Multiple Bandwidths



Helix Universal Server and RealOne Player can even adjust the bandwidth choice to compensate for network conditions. If a fast connection becomes bogged down because of high network traffic, Helix Universal Server switches to a lower-bandwidth stream to prevent the presentation from stalling. When the congestion clears, Helix Universal Server switches back to the higher-bandwidth stream. The following illustration shows a SureStream clip streaming to a 56 Kbps modem. It begins streaming at 34 Kbps, downshifts to 20 Kbps during network congestion, then upshifts to 34 Kbps when the congestion clears.

SureStream Clip Streaming at a Slower Rate During Network Congestion



You can choose SureStream and specify many different target audiences when encoding a RealAudio or RealVideo clip with Helix Producer. Note, though, that SureStream does not work when delivering a RealAudio or RealVideo clip with a Web server. If you are using Web server delivery, you can encode each clip for only one target audience. You can choose the Web server delivery option through the Helix Producer user interface when encoding your clips.

Tip: You can also use SMIL to stream different presentations based on viewer bandwidth. After you learn the basics of SMIL in this guide, refer to the switching chapter of *RealNetworks Production Guide*.

Video Dimensions

An encoder such as Helix Producer typically can turn your source video file into a clip that streams to any target connection. But if the encoder has to squeeze a file down too much to reach a low-bandwidth target, clip quality may degrade. So although the clip will stream well, you might not like the visual results. To ensure high-quality playback, select a video height and width that are appropriate for your target audience. You can set the clip size with your video editing tool. When encoding in the RealVideo format, you can also resize or crop a video when encoding it with Helix Producer.

The following table lists four common video clip dimensions that maintain the 4:3 aspect ratio used in television. For each clip size, the table indicates the general playback quality you'll get when streaming a video clip to various

target audiences. “Excellent” video quality means few visual imperfections and a video frame rate that results in acceptably smooth playback. Lower-quality video playback means more imperfections and a lower frame rate.

RealVideo Quality at Different Dimensions and Bandwidths

Target Audience	Clip Speed	Video Quality for Dimensions in Pixels			
		176 x 132	240 x 180	320 x 240	640 x 480
28.8 Kbps modem	20 Kbps	Excellent	Good	Fair	Poor
56 Kbps modem	34 Kbps			Good	
64 Kbps single ISDN	45 Kbps		Good		
112 Kbps dual ISDN	80 Kbps		Excellent	Excellent	Fair
Corporate LAN	150 Kbps				Good
256 Kbps DSL/cable	225 Kbps		Excellent	Excellent	Excellent
384 Kbps DSL/cable	350 Kbps				
512 Kbps DSL/cable	450 Kbps				

The table shows that when streaming larger videos, you get good to excellent quality only at higher connection speeds. Your results will also vary based on the streaming format. RealVideo 9, for example, produces better quality than the RealVideo 8 format. If you plan to stream clips over modems, you can first encode your clip at 320 by 240 pixels to test its quality. If you want better quality, shrink the video dimensions with your editing software, or with Helix Producer during encoding.

Tip: When resizing a video with Helix Producer, setting the RealVideo preferences to high-quality resize rather than a fast resize yields better results.

When to Lower the Streaming Speed

In some cases, you may want your clips to stream at less than the maximum recommended streaming speed listed in the table “Maximum Streaming Rates” on page 25. In the following cases, you may want to reduce your clip’s bandwidth:

- If your clip or presentation opens up large HTML pages in the RealOne Player media browser or related info pane, you may need to leave some bandwidth available so that RealOne Player can download the pages at a

reasonable rate. This is especially true when delivering a presentation over a modem.

- You may need to leave enough bandwidth for the user to perform other network activities. When streaming an Internet radio station, for example, leave some bandwidth for the listener to view Web pages.
- When several clips play together, their streaming speeds added together should not exceed the connection maximum. For example, a Flash clip and a RealAudio clip streaming at 12 and 8 Kbps, respectively, can play in parallel over 28.8 Kbps modems because together they stream at 20 Kbps. However, they cannot play back together if they stream at 12 and 16 Kbps, respectively, because the 28 Kbps total streaming speed leaves the modem no bandwidth for overhead. Such a presentation would likely pause often.
- Bandwidth is shared by everyone on a local area network (LAN). If the LAN is heavily used, the 150 Kbps LAN target speed may slow down the LAN too much. For an intranet, the LAN manager should decide the maximum streaming rate.

Delivery Options

How will you deliver your clips to other people? As the following sections explain, how you plan to stream your clips can greatly affect your media production.

Helix Universal Server Streaming

Helix Universal Server is the preferred host for streaming presentations. Designed specifically to stream multimedia over networks, Helix Universal Server keeps multiple clips synchronized and uses many advanced features to ensure that clips stream smoothly, even under adverse network conditions. A Helix Universal Server administrator sets up and runs each Helix Universal Server. If you will not be running Helix Universal Server yourself, check the following with your Helix Universal Server administrator:

1. What server version is available?

To deliver clips described in this manual, you'll need RealSystem Server 8, or Helix Universal Server 9. Make sure that your Helix Universal Server can deliver all of the clips that you plan to develop.

2. How many streams can Helix Universal Server deliver?

Each Helix Universal Server has a maximum number of media streams it can send out at once. A Helix Universal Server with a maximum of 500 streams, for example, can stream video to 500 viewers simultaneously. Make sure that the Helix Universal Server you plan to use has adequate capacity for your needs.

3. Are there any bandwidth constraints?

The Helix Universal Server computer may lack the outgoing bandwidth to deliver a lot of high-speed clips simultaneously. If you plan to develop high-bandwidth presentations, confer with the Helix Universal Server administrator about bandwidth limitations.

4. Where will your clips reside?

Your clips typically reside on Helix Universal Server, whereas your Web pages are on a Web server. You'll need to know the URLs for your clips on Helix Universal Server so that you can set up your Ram file URLs and Web page hyperlinks correctly. (Chapter 3 covers these topics in detail.)

5. Do any Helix Universal Server features need to be set up?

The Helix Universal Server administrator can set up many streaming and security features, such as:

- live broadcasts
- pay-per-view content
- automatic ad insertion
- password authentication

Using Helix Universal Server Through an Internet Service Provider

If an Internet service provider (ISP) hosts your Web pages, contact the ISP administrator to check out the Helix Universal Server issues described above. Also find out how much disk space you will have for streaming media. Many ISPs allot you a certain amount of disk space on their servers, such as 5 or 10 Megabytes. Although this is a generous amount for Web pages, it's not much for streaming media. A single video clip can easily take up that much space.

Web Server Downloading

Although Web servers can deliver some streaming clips, they don't have Helix Universal Server's ability to synchronize clips and keep long presentations

flowing smoothly. When only a Web server is available, you can still deliver multimedia presentations, but you will not be able to use all of the features that Helix Universal Server offers, such as SureStream technology.

For More Information: *RealNetworks Production Guide* has a section on Web server limitations in its presentation delivery chapter.

THE RAM FILE

The Ram file is the basic means for launching a clip or a simple sequence of clips that plays in RealOne Player. You can also use the Ram file to open an HTML page in the media browser or related info pane. This chapter explains the Ram file syntax, covers the URLs to use with streaming clips, and shows you how to pass basic parameters to RealOne Player.

Launching Clips with a Ram File

When your presentation is ready to stream, you write a Ram file, which is so-named because it uses the file extension `.ram`, as in `my_presentation.ram`. Also known as a *metafile*, the Ram file links your Web page to your clip. Your Web page links to the Ram file with a standard `<a href>` tag, and your Ram file contains the full URL to your streaming presentation. So instead of linking directly to your clip, your Web page links to the Ram file, which, in turn, links to the clip.

Is a Ram File Necessary?

The Ram file may seem like an unnecessary middle step, but it's important for the following reasons:

1. The Ram file launches RealOne Player.

The file extension `.ram` causes a Web browser to launch RealOne Player to play the presentation. RealOne Player might not launch when you link directly to a clip. When you link your Web page directly to a Flash Player file (extension `.swf`), for example, the browser launches Macromedia's Flash Player. If you intend to stream your Flash clip, you need to use a Ram file to launch RealOne Player instead.

2. The Ram file provides an RTSP URL for clips on Helix Universal Server.
Clips on Helix Universal Server stream over the RTSP protocol, rather than HTTP. This means that the URL used to request the clips must start with `rtsp://` rather than with `http://`. Because browsers cannot make RTSP requests, you link your Web page to a Ram file with an HTTP URL. The Ram file then gives RealOne Player the RTSP URL to your presentation.
3. The Ram file can pass parameters to RealOne Player.
Through these parameters, you can specify the HTML page or pages you want to display in RealOne Player's HTML panes as the clip plays.

The only time you shouldn't use a Ram file to launch RealOne Player is when you're using the Ramgen feature of Helix Universal Server, which lets you link your Web page directly to a streaming media clip. Even though your Web page link uses HTTP, the clip will stream to RealOne Player using RTSP. Ramgen doesn't offer all of the features of a Ram file, but it provides a simple way to link clips directly to your Web page.

For More Information: For more information about using Ramgen, see the presentation delivery chapter of *RealNetworks Production Guide*. Your *Helix Universal Server Administration Guide* explains how to set up Ramgen.

Writing a Ram File

The most basic Ram file has only one line: the full URL to a clip. You can write a Ram file with any text editor, word processor, or HTML editor that can save output as plain text. On Windows, you can use Notepad. On the Macintosh, Simple Text will work fine.

► To write a Ram file:

1. Open a new, blank file in your text editor. On the top line, enter the full URL to your media clip or SMIL presentation. The following are some sample URLs depending on the type of server used to deliver the clip:

Helix Universal Server:	<code>rtsp://helixserver.example.com/my_video.rm</code>
Web server:	<code>http://www.example.com/my_video.rm</code>
local:	<code>file://my_video.rm</code>

For More Information: For more about URLs, see “What URLs Do You Use?” on page 35.

Tip: Don’t use spaces in clip file names or the Ram file name. Although many servers can stream files that include spaces in their names, excluding spaces can help prevent problems during streaming.

2. If you want to stream a series of clips or SMIL presentations in sequence, simply list the URL to each clip on a separate line. Here’s an example:

```
rtsp://helixserver.example.com/video1.rm
rtsp://helixserver.example.com/video2.rm
rtsp://helixserver.example.com/video3.rm
```

Warning! Press **Enter** only to create a new line on which you want to enter a new URL. Do not press **Enter** when typing in a long URL. It’s OK if your text editor wraps the URL to a new line automatically, though. Only a line break you enter yourself will cause an error.

3. Save the Ram file as plain text with a .ram extension.

Windows text editors such as Notepad automatically apply the .txt extension to files. To avoid this, choose “All Files” as the file type in the **File>Save As** dialog. To edit your Ram file later, you must first open the text editor, then use the editor’s **Open** command to open the Ram file because double-clicking the file launches RealOne Player

Tip: If your file is saved with a .txt extension, you can remove the extension or change it to .ram on your computer desktop.

What URLs Do You Use?

Although the Ram file is simple to write, it’s often the source of the biggest problem people face with streaming: using an incorrect URL. There are two important things to keep in mind when writing a Ram file:

1. The protocol must match the server.
 - a. A Web server uses the HTTP protocol, so URLs to clips on it must start with:
http://

- b. Helix Universal Server uses the RTSP protocol, so URLs to clips on it must start with:
rtsp://
 - c. To test the clip on your computer first, place the Ram file in the same directory as the clip. In the Ram file, use just the clip name preceded by the local URL indicator:
file://
2. The URL must indicate the correct path to the clip. If you are not running the Web server or Helix Universal Server yourself, you'll need to get the path to the clip from the server administrator. Keep in mind that the path in the URL typically does not reflect the actual operating system path on the server machine.

Why Does Helix Universal Server Use RTSP?

To deliver HTML pages and graphics, a Web server uses Hyper Text Transport Protocol (HTTP), which downloads files without regard to timelines, making clips with timelines more likely to stall. Although Helix Universal Server can also use HTTP, it generally uses Real-Time Streaming Protocol (RTSP) to deliver streaming clips. RTSP is an Internet standard set forth by the Internet Engineering Task Force (<http://www.ietf.org/>).

Designed for streaming, RTSP enables Helix Universal Server to keep clips playing smoothly. As a clip streams, RealOne Player communicates with Helix Universal Server about the clip's progress, indicating how much data it needs to keep playback flowing smoothly. Helix Universal Server can then adjust the data flow to compensate for changing network conditions, reducing low priority data if necessary to ensure that crucial data gets through. Communication like this is not possible through HTTP.

Putting Comments In a Ram File

You can place a comment to a Ram file by using a pound sign (#) as the first character on a line. The following example shows a Ram file with two comment lines:

```
# Three videos that play in sequence.  
# Total playing time is 10 minutes, 40 seconds.  
rtsp://helixserver.example.com/video1.rm  
rtsp://helixserver.example.com/video2.rm  
rtsp://helixserver.example.com/video3.rm
```

Adding Parameters to Your Ram File

A Ram file provides a simple and convenient way to set parameters that open HTML pages in the RealOne Player related info and media browser panes. Ram file parameters can also affect the media by shortening a clip's playback time, for instance. In the Ram file, separate the first parameter from the clip URL with a question mark (?), as shown here:

URL?parameter=value

To set two or more parameters for the same clip, precede the second and all subsequent parameters with ampersands (&) instead of question marks:

URL?parameter=value¶meter=value¶meter=value...

Note the following about Ram file parameters:

- Parameter values that contain spaces need to be enclosed in double quotation marks. Single values do not require quotation marks, though.
- Do not press **Enter** to create a line break when adding parameters to a clip URL. The presentation URL and all parameters must be on a single line. You can turn your text editor's word wrap feature on, though, so that the line wraps automatically.
- Previous versions of RealPlayer that do not support a specific parameter will ignore the parameter and still play the media.

Tip: Appendix D summarizes the Ram file parameters that the following sections describe in detail.

Opening an HTML Page as a Clip Plays

For each clip in the Ram file, you can provide the URL to one HTML page that opens in the RealOne Player related info pane. You can also provide a URL to

an HTML page that opens in the media browser pane. The following table lists the Ram file HTML page parameters.

Parameters for Opening HTML Pages as a Clip Plays

Parameter and Value	Function
<code>rpcontexturl=URL _keep</code>	Displays the specified URL in the related info pane, or keeps the existing related info pane open. Use a fully qualified HTTP URL. If testing with a local clip, use the full, absolute path to the clip on your computer.
<code>rpcontextheight=pixels</code>	Sets the pixel height of the related info pane. If no height is specified, RealOne Player uses the height of the media clip. See “Related Info Pane Sizing” on page 12 for more information.
<code>rpcontextwidth=pixels</code>	Sets the pixel width of the related info pane. If no width is specified, a default of 330 pixels is used.
<code>rpcontextparams=URL_parameters</code>	Appends parameters to the <code>rpcontexturl</code> URL. HTML page parameters are generally separated from the page URL with a question mark. In a Ram file, however a question mark indicates the start of the Ram file parameters. Hence, if you need to append parameters to your related info page URL, do so through <code>rpcontextparams</code> .
<code>rpcontexttime=dd:hh:mm:ss.x</code>	Specifies the time at which the HTML page displays in the related info pane, relative to the start of the media clip. Only the seconds (ss) field is required, so <code>rpcontexttime=10</code> means to open the related info pane 10 seconds after the clip starts to play. If no time is specified, the page opens as soon as the clip starts to play. Use of <code>start=hh:mm:ss.x</code> with the clip does not affect when the HTML page displays.
<code>rpurl=URL</code>	Specifies the URL to display in the media browser pane. This URL always opens when the clip begins to play. If testing with a local clip, use the full, absolute path to the clip on your computer.
<code>rpurlparams=URL_parameters</code>	Appends parameters to the <code>rpurl</code> URL. If you need to add parameters to your media browser page URL, do so through <code>rpurlparams</code> .

(Table Page 1 of 2)

Parameters for Opening HTML Pages as a Clip Plays (continued)

Parameter and Value	Function
<code>rpurltarget=_rpbrowser name</code>	Sets the target for <code>rpurl</code> as the media browser pane when you use <code>_rpbrowser</code> , or as a secondary browsing window if you use any other name. Because the default is <code>_rpbrowser</code> , you can omit this parameter to use the media browser.
<code>rpvideofillcolor= color_value</code>	Specifies a background color for the media playback pane, allowing you to match the backgrounds of the media playback and related info panes. Black is the default color. See below for more about colors.

(Table Page 2 of 2)

Tip: To open more than one HTML page for a clip in the related info pane or media browser pane at any point during the presentation, you can write a SMIL file as described in Chapter 6.

Background Color Values

For `rpvideofillcolor`, you can use one of 16 predefined color names, an RGB value, or a six-digit hexadecimal value. Illustration programs typically define colors using the RGB scheme, while the hexadecimal scheme is common to HTML markup programs. The following table lists the predefined names, along with that color's corresponding hexadecimal and RGB values.

white #FFFFFF rgb(255,255,255)	silver #C0C0C0 rgb(192,192,192)	gray #808080 rgb(128,128,128)	black #000000 rgb(0,0,0)
yellow #FFFF00 rgb(255,255,0)	fuchsia #FF00FF rgb(255,0,255)	red #FF0000 rgb(255,0,0)	maroon #800000 rgb(128,0,0)
lime #00FF00 rgb(0,255,0)	olive #808000 rgb(128,128,0)	green #008000 rgb(0,128,0)	purple #800080 rgb(128,0,128)
aqua #00FFFF rgb(0,255,255)	teal #008080 rgb(0,128,128)	blue #0000FF rgb(0,0,255)	navy #000080 rgb(0,0,128)

With a hexadecimal color value, substitute the escape character `%23` for the pound sign (`#`), which, in a Ram file, signifies the start of a comment. For example, suppose that you want to match the following hexadecimal color used in a related info HTML page:

```
<BODY BGCOLOR="#FF5A4E">
```

You would add the following to your Ram file:

```
rpvideofillcolor=%23FF5A4E
```

Examples of Opening HTML Pages

Opening a Page in the Related Info Pane

The following example plays a clip and opens an HTML page in a related info pane that is 250 pixels high and 280 pixels wide:

```
rtsp://helixserver.example.com/video1.rm?rpcontextheight=250  
&rpcontextwidth=280&rpcontexturl="http://www.example.com/relatedinfo1.html"
```

Opening a Page in the Media Browser Pane

The next example opens an HTML page in the media browser pane when the clip begins to play:

```
rtsp://helixserver.example.com/video2.rm?rpurl="http://www.example.com/index.html"
```

Keeping the Same Context Pane, But Changing Background Colors

The following sample Ram file plays two clips in sequence. After the first clip plays for 5.5 seconds, the Ram parameters open an HTML page in a related info pane that is 350 pixels high by 300 pixels wide. The media playback pane's background color is set to rgb(30,60,200). When the second clip plays, the same related info pane is kept onscreen, but the media playback pane's background changes to red:

```
# First URL that opens a related info pane.  
rtsp://helixserver.example.com/video3.rm?rpcontextheight=350  
&rpcontextwidth=300&rpcontexturl="http://www.example.com/relatedinfo2.html"  
&rpcontexttime=5.5&rpvideofillcolor=rgb(30,60,200)  
#  
# Second URL that keeps the same related info pane,  
# but changes the media playback pane's background color.  
rtsp://helixserver.example.com/video4.rm?rpcontexturl=_keep  
&rpvideofillcolor=red
```

Controlling How a Presentation Initially Displays

In the Ram file, you can set several parameters that control how RealOne Player initially displays a clip or SMIL presentation. You can play a clip at double its normal size, for example, play part of a clip, or open the RealOne

Player at full-screen size. To control these characteristics, add one or more of the following parameters to the Ram file URL.

Parameters for Setting the Initial Display

Parameter and Value	Function
<code>screensize=double full original</code>	Opens the clip or presentation at double its normal size, at full-screen size, in which the monitor looks like a television set, or at its original size, which is the default.
<code>mode=normal theater toolbar</code>	Opens RealOne Player in one of three modes. In normal mode, which is the default, controls are grouped around the media playback pane. In toolbar mode, which is available only to subscribers of the premium services, the controls appear at the bottom of the computer screen. In theater mode, controls are put in toolbar mode, and the media presentation appears centered on a darkened screen.
<code>start=hh:mm:ss.x</code>	Starts the clip at the specified point in its timeline. Only the seconds field is required, so <code>start=45</code> begins the clip at its 45-second mark. This parameter shortens the total time the clip plays, but it does not delay the clip from starting its playback.
<code>end=hh:mm:ss.x</code>	Ends the clip at the specified point in its timeline. Only the seconds field is required. For example, <code>end=3:30</code> means to end the clip when it reaches its internal mark of three minutes and thirty seconds. The total time that the clip plays is the end time minus the start time.
<code>showvideocontrols overlay=0 1</code>	When set to 0, hides the sizing overlay that appears briefly when the viewer moves the screen pointer over the media playback pane. (The overlay, which appears in the upper-left corner of the media playback pane, has controls to display the media at different sizes.) This parameter works only with RealOne Player version 2 and higher.

Examples of Setting a Clip's Initial Display

Opening a Clip in Full-Screen Mode

The following example opens a SMIL presentation in full-screen mode:

```
rtsp://helixserver.example.com/sample1.smil?screensize=full
```

Opening a Clip at Normal Size in Theater Mode

The next example opens a video clip at double its normal size, and sets RealOne Player to its theater mode:

```
rtsp://helixserver.example.com/video1.rm?screensize=double&mode=theater
```

Playing a Clip Excerpt

The final example plays a 30-second excerpt from the middle of a clip:

`rtsp://helixserver.example.com/audio1.rm?start=55&end=1:25`

Tips for Setting the Initial Display

- HTML panes do not display when the media plays at full-screen size. Therefore, you should not open an HTML page when also using `screensize=full`.
- You do not need to specify a mode parameter when using `screensize=full`.
- RealOne Player may not offer full-screen mode on all operating systems. If RealOne Player for a given operating system does not offer full-screen mode, it plays the presentation at its normal size.
- If RealOne Player offers full-screen mode but has not yet played a clip full-screen, it may first perform a test of this playback mode.
- The double-size and full-screen modes work best for high-speed clips. They are not recommended for presentations delivered through modems.
- Always test playback when using double and full-screen modes to ensure that the visual quality is acceptable. Some types of clips may not scale well.
- In full-screen mode, the viewer can control RealOne Player through a contextual menu displayed by right-clicking (on Windows) or holding down the mouse button (on Macintosh).
- RealOne Player displays a presentation's elapsed time in one-second increments. You can click the time-elapsed field to display time values to 1/10th of a second, however. This can help you decide what start and end timing values you want to use in a Ram file.
- For a SMIL presentation, use `clipBegin` and `clipEnd` in the clip source tag, rather than `start` and `end` in the Ram file, to play an excerpt from a clip. For more information, see "Using Internal Clip Begin and End Times" on page 76.
- The `showvideocontrolsoverlay` parameter is intended primarily for media presentations that include interactive elements, such as Flash clips or SMIL presentations. It allows you to hide the overlay so that it does not interfere with buttons or controls that are part of the media.

Overriding Title, Author, and Copyright Information

A streaming clip often encodes title, author, and copyright information. When you encode a RealAudio or RealVideo clip, for example, you can add this information to the clip through Helix Producer. Through the Ram file, you can override this title, author, and copyright information. These parameters are compatible with earlier versions of RealPlayer.

Title, Author, and Copyright Parameters

Parameter and Value	Function
<code>title="text"</code>	Specifies the clip title.
<code>author="text"</code>	Indicates the clip author. This information displays in the Artist field of the clip information panel.
<code>copyright="text"</code>	Gives the copyright notice. You can use the HTML code <code>&#169;</code> to create the standard copyright symbol.

Example of Setting Title, Author, and Copyright Information

The following example sets title, author, and copyright information for a video clip:

```
rtsp://helixserver.example.com/introvid.rm?title="Introduction to Streaming
Production"&author="RealNetworks, Inc."&copyright="&#169;2001,
RealNetworks, Inc."
```

Tips for Using Title, Author, and Copyright Parameters

- Whether information is encoded in the clip, or added through a Ram file, it appears in the following areas of RealOne Player:
 - Title, author, and copyright information displays when the viewer chooses the **File>Clip Properties>View Clip Info** command, or presses **Ctrl+i**.
 - Title, author, and copyright information crawls horizontally along the title bar at the top of RealOne Player, unless the SMIL file also uses presentation information as described in “Presentation Information” on page 61. In this case, only the presentation information appears in the title bar.
 - Title and author information appears in the “Now Playing” list, which is part of the RealOne Player media browser pane. Viewers can double-

click a clip listing to play that clip. See “Now Playing List” on page 14 for more information.

- Title information appears in the recent clips list under the RealOne Player **File** menu, unless the SMIL file also includes a presentation title. In this case, only the presentation title appears in the list.
- You can use any combination of title, author, and copyright parameters in a Ram file, but RealNetworks highly recommends that you always include title parameters for clips that have no encoded titles. If no title is available, a clip’s file name displays in place of the title.
- There are similar attributes that you can set within a SMIL file instead of the Ram file. See “Adding Clip or Group Information” on page 73 for more information.

Setting Clip Information

The clipinfo parameter is new with RealOne Player, and is ignored by earlier RealPlayers. Geared for online music, it allows you to encode information such as the artist name, album, genre, and so on, which displays when the viewer chooses the **File>Clip Properties>View Clip Info** command, or presses **Ctrl+i**. The clipinfo parameter uses one long value surrounded by double quotation marks. Within the quotes, you separate the subvalues with vertical lines, or “pipes,” as shown here:

```
clipinfo="name=value|name=value|name=value..."
```

The following table describes the name and value pairs that you can use with clipinfo. You can use any set of values, and list them in any order. Most text values can be over 100 characters long.

Clipinfo Parameter Values

Name and Value	Function
title= <i>text</i>	Gives the clip title.
artist name= <i>text</i>	Indicates the artist name.
album name= <i>text</i>	Gives the album name. If you specify an album name and do not also display an HTML page in the related info pane, RealOne Player displays in that pane a standard page that lists the artist, album, year, and genre values. The viewer can hide this information, though, with Tools>Album Info>Hide .
genre= <i>text</i>	Indicates the clip genre, such as Rock or Jazz.

(Table Page 1 of 2)

Clipinfo Parameter Values (continued)

Name and Value	Function
<code>copyright=text</code>	Gives the copyright notice.
<code>year=text</code>	Indicates the year the content was released.
<code>cdnum=number</code>	Supplies the CD track number.
<code>comments=text</code>	Provides any additional comments.

(Table Page 2 of 2)

Note: Do not use the title, author, and copyright parameters described in “Overriding Title, Author, and Copyright Information” on page 43 along with clipinfo.

Using Text Escape Characters

To use certain text characters in a value for the clipinfo parameter, you must use the character’s corresponding escape code. This is because certain characters represent syntax components. A pipe (|) represents the start of a new value, for example, so to use a pipe within a value, you must use the escape code %7C. The following table lists some common text characters that you can add through escape codes.

Text Character Escape Codes

Name	Character	Escape Code
ampersand	&	%26
apostrophe	'	%27
backslash	\	%5C
carat	^	%5E
double quote	“	%22
greater than sign	>	%3E
left bracket	[%5B
less than sign	<	%3C
percent sign	%	%25
pipe		%7C
pound sign	#	%23
right bracket]	%5D

You can enter other common text characters, such as commas, periods, and colons directly into clipinfo parameter. Conversely, you can display *any* text

character, including letters and numbers, by using an escape code that starts with % followed by the character's ASCII hexadecimal value. You can create an asterisk (*) with the escape code %2A, for example.

For More Information: Visit <http://www.asciitable.com> for a full list of ASCII codes.

Example of Setting Clip Information

This example sets the clipinfo parameter for an audio clip:

```
rtp://helixserver.example.com/song1.rm?clipinfo="title=Artist of the Year|  
artist name=Your Name Here|album name=My Debut|genre=Rock|  
copyright=2001|year=2001|comments=This one really knows how to rock!"
```

The following figure illustrates how this information appears in the clip information panel (**Ctrl+i**).

Clip Information



Moving Files to a Server

When your media clips, Web pages, and Ram file are ready for delivery, transfer them to your Web server and Helix Universal Server, placing them in the directories prepared by the server administrators. If a server is on the same local area network (LAN) as your computer, you can often just copy the files to the server over the network. Otherwise, you can usually transfer files to a server over the Internet using FTP (file transfer protocol).

Using FTP to Transfer Clips

The FTP protocol is designed to copy files from one computer to another. Many computers have an FTP application preinstalled. FTP applications are also available for download from many Internet software archives. Your Helix Universal Server or Web server administrator will have to set up FTP access to the server machine for you, as well as give you an FTP user name and password.

FTP distinguishes between a “text” mode for transferring text-only files, and a “binary” mode for transferring non-text files such as streaming media clips. Some FTP programs can set this mode automatically. With other programs, though, you must set the transfer mode yourself. If you transfer streaming clips in text mode, the clip may become corrupt. Fortunately, FTP never modifies the original clips on your computer, so you can simply transfer the clips again using the binary mode.

Tip: Helix Producer and RealSlideshow can transfer files to a server automatically. Refer to their user manuals or online help for more information.

Where Does the Ram File Go?

Move your Ram file to Helix Universal Server or your Web server. Even if all of your media clips are on Helix Universal Server, you can place the Ram file on your Web server or any other server. Your Web page, Ram file, and media clips do not have to reside together. If you keep the Ram file in the same directory as your main Web page, you can use a relative link like the following:

```
<a href="play_video1.ram">Play the video!</a>
```

If the Ram file is in a different location, use a full HTTP URL in the link to the Ram file:

```
<a href="http://www.example.com/play_video1.ram">Play the video!</a>
```

Then, make sure that the Ram file gives RealOne Player either the full RTSP URL to the clip on Helix Universal Server:

```
rtsp://helixserver.example.com/video1.rm
```

or the full HTTP URL to the clip on a Web server:

```
http://www.example.com/video1.rm
```


CLIP-ENCODED URLs

When you produce RealAudio or RealVideo clips using Helix Producer, you can use a utility that embeds clip information and HTML URLs directly in the clip. This lets you add many of the features available through a Ram file directly to your streaming audio or video clip.

For More Information: See *Helix Producer User's Guide* for information about additional parameters that you can encode into a clip. You can turn a RealVideo clip into an image map, for example. You can download Helix Producer from <http://www.realn networks.com/products/producer/index.html>, and display the user's guide from the **Help** menu.

Using Clip-Encoded URLs

The **RMEvents** utility runs on all Windows operating systems, as well as Linux and the Macintosh. Any version of **RMEvents** can encode the information described in this chapter, so you can use a utility that predates RealOne Player, such as the one that comes with RealSystem Producer 8.5. The URL-encoding method is backwards-compatible with earlier versions of RealPlayer. If a viewer has RealPlayer 8, for example, the encoded URLs display in the viewer's default browser.

Because this production technique encodes URLs directly into the clip, it is not recommended if you want the HTML pages associated with clips to change. (**RMEvents** preserves the original copy of your clip, though, so you can always change the URLs by running **RMEvents** again.) Additionally, it's easier to use SMIL links, as described in Chapter 6, if your RealAudio or RealVideo clip will be part of a multiclip presentation.

Writing an Events File

To use **RMEvents**, you first create an events file, which is a plain text file that uses the standard text extension of `.txt`. This file describes events that occur as the clip plays. You write each event on a separate line, and you can use a pound sign (`#`) to start a comment line. Each event line follows this format:

```
flag start_time end_time event_syntax
```

The flag indicates the type of event, which can be either to display clip information, or to open a URL in an HTML pane automatically. The starting time and ending time are relative to the start of clip playback. You indicate the time value with the following format, in which only the seconds field is required:

```
dd:hh:mm:ss.x
```

Note: As with a Ram file, define each event on a single line within the events text file. Do not press **Enter** to wrap long lines manually.

Specifying URL Events

When opening a URL automatically in an HTML pane, you use the `u` event flag. The event syntax looks like this:

```
u start_time end_time &&target&&URL?parameters
```

The URL must be a fully qualified HTTP URL. For the parameters, you can use `rpcontextheight` and `rpcontextwidth` when sending a URL to the related info pane. For *target*, use one of the following:

<code>_rpcontextwin</code>	Display the URL in the related info pane.
<code>_rpbrowser</code>	Display the URL in the media browser pane.
<code>_rpexternal</code>	Display the URL in a secondary browsing window.

Tip: The `rpcontextheight` and `rpcontextwidth` parameters are described in the table “Parameters for Opening HTML Pages as a Clip Plays” on page 38. You cannot use `rpvideofillcolor` as a parameter.

The following is a sample events file that opens two URLs in the related info pane at different times, and two URLs in the media browser pane at different times:

```

# Open a URL in the related info pane when the clip starts, and size the pane.
u 00:00:00.0 00:01:59.9 &&_rpcontextwin&&
http://www.example.com/info1.html?rpcontextheight=250&rpcontextwidth=280
#
# Open a URL in the media browser pane at the 1-minute mark.
u 00:01:00.0 00:01:59.9 &&_rpbrowser&&http://www.example.com/index.html
#
# Open a second URL in the related info pane at the 2-minute mark.
u 00:02:00.0 00:04:00.0 &&_rpcontextwin&&http://www.example.com/info2.html
#
# Open a second URL in the media browser pane at the 3-minute mark.
u 00:03:00.0 00:04:00.0 &&_rpbrowser&&http://www.example.com/index2.html

```

Note the following about this sample:

- When you open multiple URLs, list the events in ascending order according to the start times.
- The related info pane size is set for the duration of the clip by the first URL that opens in that pane. Any subsequent URLs that target the related info pane therefore do not require sizing information.
- The end times indicate a point past which the URL should not open. In the second event defined above, the URL is scheduled to open at 1:00 minutes, but no later than 1:59.9 minutes. If a viewer starts the clip and immediately seeks to its 3-minute mark, for example, the URL doesn't display because the clip never plays at any point between the URL's start and end times.

Adding Clip Information

Using **RMEvents**, you can encode the Ram file clipinfo parameters described in “Setting Clip Information” on page 44. As with a Ram file, separate the subvalues with vertical lines, or “pipes.” The following example shows an events file line that specifies clip information, using `i` as the event flag:

```
i 00:00:00.0 00:00:10.0 clipinfo:title=My Presentation|artist name=Pat Morales|...
```

For clip information, you can specify the start time as the clip's starting time (00:00:00.0). Specifying an end time is required, but the actual end time doesn't matter because the clip information will display throughout the length of the clip playback. In the example above, the end time is set to 10 seconds after the clip starts.

Note: You can also encode into a clip the title, author, and copyright information described in “Overriding Title, Author, and Copyright Information” on page 43. Typically, you add this information through the Helix Producer interface when encoding a clip, but you can also add it later through the **RMEvents** utility. For more information, see *Helix Producer User’s Guide*.

Merging the Events File with the Clip

After you write your events file, you use the **RMEvents** utility to merge the events file with your clip. The following are instructions for doing this on Windows operating systems. For instructions on running **RMEvents** on Linux or the Macintosh, as well as for information about additional command options, refer to your *Helix Producer User’s Guide*.

► To merge an events file with a RealAudio or RealVideo clip on Windows:

1. Open a command line prompt from the **Start** menu.
2. Move to the main Helix Producer directory. The default main directory is C:\Program Files\Real\Helix Producer Basic or C:\Program Files\Real\Helix Producer Plus.

Here is an example:

```
C:\>cd "C:\Program Files\Real\Helix Producer Basic"
```

3. Enter the following command, which uses three flags to indicate the input clip, output clip, and events file:

```
rmevents -i input.rm -o output.rm -e events.txt
```

where:

- *input.rm* is the path and name of the input clip
- *output.rm* is the path and name of the output clip
- *events.txt* is the path and name of the events file

Tip: Always choose a new output name so that you can save your original clip without any encoded events.

Streaming the Clip

After you have encoded events into your RealAudio or RealVideo clip, you can stream the clip by transferring it to your server and linking it to a Web page through a Ram file, as described in Chapter 3. Note that you can also add parameters to the Ram file. If you have encoded HTML URLs or clipinfo information into your clip, you shouldn't also use these parameters in the Ram file. But you may want to use other Ram file parameters, such as `screenSize=double`, for instance, to open the clip at twice its encoded size.

JAVASCRIPT COMMANDS

Using Javascript commands, you can enhance RealOne Player presentations. This chapter describes commands that you can use to play a clip in the media playback pane, set the media playback pane's background color, and open an HTML page in the related info or media browser pane. Although these methods are simple, a familiarity with Javascript is recommended.

Note: The methods described in this chapter are intended only for HTML pages that display in the related info and media browser panes, not for HTML pages rendered by another Web browser.

Tip: For samples of the PlayClip() method in action, download the zipped HTML version of this guide as described in “How to Download This Guide to Your Computer” on page 3, copy the entire samples folder to your C: drive, and view the HTML page in the interactive folder.

For More Information: For instructions about using all of the Javascript and VBScript methods available for RealOne Player, see *RealOne Player Scripting Guide*.

Declaring Javascript Methods

To use the methods described in this chapter, you must declare them in the script section of the HTML page displayed in the media browser pane or the related info pane. Here is an example:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function RealOneLink(URL,clipinfo,context_URL,width,height){
    window.parent.external.PlayClip(URL,clipinfo,context_URL,width,height)
}
```

```
function BrowserLink(URL){
    window.parent.external.OpenURLInPlayerBrowser(URL)
}
function BackgroundColor(color){
    window.parent.external.SetVideoBackgroundColor(color)
}
-->
</SCRIPT>
```

Playing a Clip

The `PlayClip()` method allows you to create a hypertext link that opens a clip in the media playback pane. Optionally, it can pass clip information, open HTML pages in the related info and media browser panes, and set the size of the related info pane. A Ram file can perform all of these actions, too, but using a Ram file displays a file download dialog when RealOne Player requests the file. Using the Javascript `PlayClip()` method prevents this dialog.

PlayClip() Arguments

The `PlayClip()` method uses the following arguments. Only the URL argument is required to specify a clip to play in the media playback pane.

URL

The URL argument provides the fully-qualified URL to the media clip. For testing, you can use an absolute, local URL (such as `file:///C:/temp/clip.rm`), but not a relative, local URL.

clipinfo

This argument contains a string of clip information that the viewer can display by choosing the **File>Clip Properties>View Clip Info** command, or pressing **Ctrl+i**. It takes the same values as the Ram file `clipinfo` parameter, which is described in “Setting Clip Information” on page 44. Separate the subvalues with vertical lines, or “pipes,” as shown here:

```
'title=My Presentation|artist name=Pat Morales|...'
```

context_url

This argument gives the fully-qualified URL to the HTML page to display in the related info pane as soon as the clip starts to play.

width and height

These two arguments set the width and height of the related info pane in pixels. For more on related info pane sizing, see “Related Info Pane Sizing” on page 12.

media_browser_url

This argument gives the fully-qualified URL to the HTML page to display in the media browser pane as soon as the clip starts to play.

Opening a Page in the Media Browser Pane

You can use the `OpenURLInPlayerBrowser()` method in the related info pane to open an HTML page in the media browser pane. This method uses only a URL argument, which provides the fully-qualified URL to an HTML page. For testing, you can use an absolute, local URL (such as `file:///C:/temp/page1.html`), but not a relative, local URL.

Setting the Media Background Color

The `SetVideoBackgroundColor()` method sets a color that displays behind the clip playing in the media playback pane. This background color appears if you set the related info pane to a height greater than the playing clip. Black is the default background color. The method takes a color value as an argument. You can specify colors in an RGB format, as in `rgb(120,40,87)`, or as hexadecimal values, as in `#AF543C`.

SMIL HYPERLINKS

This chapter introduces you to SMIL, explaining how to write a SMIL file that plays a single clip and opens HTML pages at various points during the presentation. Later chapters explain how to use SMIL to play sequences of clips, and to play multiple clips at the same time.

For More Information: You should also read Appendix B, which provides important information about SMIL syntax. Appendix C summarizes the SMIL attributes and values described in this guide.

What is SMIL?

Pronounced “smile,” SMIL is a markup language endorsed by the World Wide Web Consortium, which operates a Web site at <http://www.w3c.org>. SMIL lets you create simple to highly complex media presentations by coordinating any number of media clips. Using SMIL, you can replicate almost all of the features that you can set through the Ram file. Plus, SMIL offers some advanced capabilities for opening HTML pages at different points during a presentation. A SMIL presentation works in RealOne Player on any operating system.

Note: RealOne Player supports both SMIL 1.0 and SMIL 2.0. Only SMIL 2.0 can open URLs automatically in the RealOne Player HTML panes, though. Because earlier RealPlayers do not support SMIL 2.0, the SMIL authoring techniques described in this chapter work only in RealOne Player.

Writing a SMIL File

Just like a Ram file, a SMIL file is a simple text file that you can create with any text editor. As with a Ram file, make sure that you save your output as

plain text. Your SMIL file should have a simple name without spaces, and end with the extension `.smil`, as in `my_smilfile.smil`.

How Does the SMIL File Fit In?

When you use a SMIL file, your presentation functions like this:

1. You link your Web page to a Ram file with a standard `<a href>` hypertext link and an HTTP URL.

In many cases, the SMIL file will give all the presentation information, and you will use the Ram file just to launch the presentation. You still need the Ram file because SMIL is a Web standard, so its `.smil` extension may be associated with more than one media player. When you produce a presentation specifically for RealOne Player, the Ram file's `.ram` extension ensures that RealOne Player launches for all viewers.

2. Your Ram file gives the full URL to the SMIL presentation.

Because the SMIL file gives full URLs to your clips, it can reside on your Web server or on Helix Universal Server. It may reside in the same server directory as the Ram file, or it may be on a different server entirely. For information about writing a Ram file, see Chapter 3.

3. The SMIL file gives the full URLs to your streaming clips.

Clips typically reside on Helix Universal Server, and stream over the RTSP protocol. "Clip Source Tags" on page 62 explains how to request a clip through a SMIL file.

SMIL File Basics

The following is a simple SMIL file that plays a video and sets presentation information. Once you understand the basic structure of this SMIL sample as described in the following sections, you'll be ready to learn about the features that make SMIL so powerful:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <meta name="title" content="My First SMIL File"/>
    <meta name="author" content="Pat Morales"/>
    <meta name="copyright" content="(c)2001 Spectacular Media Limited"/>
  </head>
```

```

<body>
  <video src="rtsp://helixserver.example.com/video1.rm"/>
</body>
</smil>

```

SMIL File Sections

A SMIL file starts with a `<smil>` tag and ends with a `</smil>` tag. The opening tag includes two `xmlns` attributes called *namespaces*. These attributes function to identify the file as SMIL 2.0, and also to indicate that the file uses the RealNetworks customizations. The section “The SMIL Tag” on page 117 provides more detail about these namespaces. Just make sure that your `<smil>` tag contains these namespaces **exactly** as they are shown above.

Between the opening and closing tags, the SMIL file breaks down into two major sections, just like an HTML page. The optional header section, defined between `<head>` and `</head>` tags, provides presentation information and can define a layout, which this guide covers in Chapter 8. The mandatory body section, falling between `<body>` and `</body>` tags, lists all the clips that play in the presentation.

Presentation Information

The header section in the preceding example contains three `<meta/>` tags that are not necessary, but are highly recommended for every SMIL presentation. Just copy these tags to every SMIL file you write, changing the values for the content attributes as appropriate:

```

<head>
  <meta name="title" content="My First SMIL File"/>
  <meta name="author" content="Pat Morales"/>
  <meta name="copyright" content="(c)2001 Spectacular Media Limited"/>
</head>

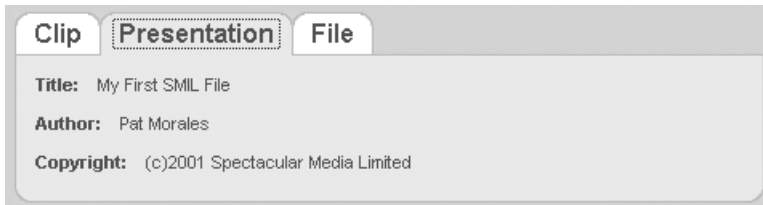
```

The `<meta/>` tags give the presentation’s title, author name, and copyright notice. They are known as “meta” tags, because their information does not display in the media playback pane along with your streaming clips, but shows up elsewhere in RealOne Player:

- Title, author, and copyright information crawls horizontally along the title bar at the top of RealOne Player.

- Title and author information appears in the “Now Playing” list, which is part of the RealOne Player media browser pane. For more on this list, see “Now Playing List” on page 14.
- Title, author, and copyright information displays when the viewer gives the **File>Clip Properties>View Clip Info** command, as illustrated in the following figure.

Presentation Information



Clip Source Tags

For each media clip in a presentation, you write a clip source tag. In the preceding sample, there is one source tag:

```
<video src="rtsp://helixserver.example.com/video1.rm"/>
```

Each clip source tag includes a `src` attribute that tells RealOne Player where to find the clip. As you’ll see later in this guide, a clip source tag can include other attributes that affect the clip’s timing, layout, and so on. Note that the preceding clip source tag closes with a slash:

```
<tag...attributes.../>
```

This is an important SMIL requirement. Leaving a closing slash out of tag that is not part of a tag pair causes an error. The following table lists some of the clip source tags that you can use.

Common Clip Source Tags

Clip Tag	Used For
<code><audio/></code>	audio clips, such as RealAudio (.rm)
<code></code>	JPEG (.jpg), GIF (.gif), or PNG images (.png)
<code><ref/></code>	any other clip type, such as a Flash Player file (.swf)
<code><video/></code>	video clips, such as RealVideo (.rm)

Although SMIL includes other clip source tags, it actually doesn’t matter which tags you use. RealOne Player never relies on a `<video/>` tag alone to

determine that a certain clip contains video. It uses the file extension, such as .rm. And even if that extension is incorrect for some reason, RealOne Player can generally determine the clip type and play the clip properly. So although you could use <ref/> tags for all clips, using the appropriate clip tags for a certain type of media helps you to keep the structure of your SMIL file clear.

Clip URLs

As you develop your presentation, keep your clips in the same directory as your SMIL presentation. That way, you can simply use the clip name exactly as it appears in your operating system as the value for the src attribute:

```
<video src="video1.rm"/>
```

When you are ready to stream your presentation, you'll need to change the src value to give the location of the clip on a streaming server, and specify the protocol used to stream the clip:

```
<video src="rtsp://helixserver.example.com/video1.rm"/>
```

For More Information: See “What URLs Do You Use?” on page 35 for more on streaming protocols.

Creating Relative Links to Other Directories

RealOne Player can also follow the same relative, local links that you can use in a Web page. This syntax can be useful as you develop your presentation on your desktop computer. For example, the following src attribute specifies a clip that resides one level below the SMIL file in the audio folder:

```
<audio src="audio/song1.rm"/>
```

The following example specifies a clip that resides one folder level above the SMIL file:

```
<audio src="../song1.rm"/>
```

The next example creates a link to a clip that resides in an audio folder that is at the same level as the folder that contains the SMIL file:

```
<audio src="../audio/song1.rm"/>
```

Tip: You can find additional information about relative directory syntax in an HTML reference guide.

Writing Absolute Links

Alternatively, you can use local, absolute links to specify exact locations on your computer. The syntax for absolute links is the same as with HTML. It varies with operating systems, however, and you should be familiar with the directory syntax for the system you are using. For example, the following absolute link syntax works for Windows computers, but not on Unix or the Macintosh. Note that this URL includes three forward slashes in file:///, and uses forward slashes in path names as well:

```
src="file:///c:/audio/first.rm"
```

Opening HTML Pages with SMIL

The following clip source tag demonstrates how to open an HTML page in the media browser pane when a clip starts:

```
<video src="rtsp://helixserver.example.com/video1.rm">  
  <area href="http://www.example.com" external="true" alt="My Home Page"  
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpbrowser"/>  
</video>
```

First, note that this clip source tag is different from the previous examples of clip source tags, which all closed with a slash:

```
<video.../>
```

Now, the clip source tag is a tag pair in which the first tag **does not** close with a slash:

```
<video...>  
  ...other markup...  
</video>
```

This is an important SMIL concept. Tags that are normally single tags can convert to tag pairs. This allows them to enclose other tags. In this case, the clip source tag encloses an <area/> tag that creates a hyperlink. A clip source tag can enclose any number of <area/> tags, letting you create more than one hyperlink associated with each clip.

How the <area/> Tag Works

An <area/> tag is always associated with a clip. As long as an <area/> tag is active, the viewer can click the clip to open the link URL. Typically, the <area/> tag becomes active as soon as the clip starts to play, and stays active as long as the clip appears onscreen. In other words, while the clip appears onscreen, the

viewer can click the clip to open the URL. There are two important ways that you can modify this behavior:

- You can set the link URL to open automatically. This lets you display an HTML page as soon as a clip starts to play, for example. See “Opening an HTML Page Automatically” on page 65 for details.
- You can use SMIL timing attributes to set how long an `<area/>` tag stays active, or to make it active after its associated clip starts playing. This lets you link to different HTML pages at different times, for example, or open an HTML page automatically at some point *after* a clip starts to play. “Using SMIL Timing Attributes” on page 75 explains timing attributes. See also the examples at the end of this chapter.

Using Standard Link Attributes

A standard `<area/>` tag hypertext link to an HTML page includes the attributes listed in the following table.

Standard `<area/>` Tag Attributes

Attribute and Value	Purpose
<code>alt="text"</code>	Provides alternate text. When the viewer moves the screen pointer over the link, the alt text displays in the status line above the RealOne Player media playback pane.
<code>external="true"</code>	Opens the URL in an HTML pane. This attribute is required for opening HTML pages.
<code>href="URL"</code>	Gives the fully qualified HTTP URL to the HTML page.

Opening an HTML Page Automatically

Include `actuate="onLoad"` in the `<area/>` tag to open the link URL as soon as the `<area/>` tag becomes active. If the `<area/>` tag does not contain any SMIL timing attributes, it becomes active as soon as its associated clip starts to play. Hence, `actuate="onLoad"` can open an HTML page automatically whenever a clip starts to play. Without `actuate="onLoad"`, the URL opens only when the viewer clicks the clip.

Tip: A link that uses `actuate="onLoad"` is still clickable, meaning that the viewer can reopen it after it opens automatically. If you want to prevent this, set a short link duration by using `dur="1s"`, for example, in the `<area/>` tag.

Controlling the Media Playback State

By default, the RealOne Player presentation pauses while the browser opens the link. The viewer can resume the presentation by clicking the RealOne Player **Play** button. You can also make RealOne Player stop the presentation completely, or continue playing when the link opens, with a stop or play value, respectively, for the sourcePlaystate attribute. Use sourcePlaystate="play" in the <area/> tag, for example, to keep the media playing as the HTML page opens.

Selecting the HTML Pane

The following table lists the attributes that you can use to open an HTML URL in one of the RealOne Player panes. The external="true" attribute is always required. The attributes that specify a RealOne Player pane use the rn: prefix, indicating that they are RealNetworks customizations. Using these attributes requires that you declare the RealNetworks namespace, which is described in "RealNetworks Extension Namespace" on page 118.

Attributes for Opening an HTML Link in RealOne Player

Attributes	Target
external="true"	A secondary media browser window that is detached from the three-pane environment.
external="true" rn:sendTo="_rpbrowser"	The media browser pane.
external="true" rn:sendTo="_osdefaultbrowser"	The viewer's default Web browser.
external="true" rn:sendTo="_rpcontextwin"	The related info pane.

Opening HTML Pages in the Related Info Pane

To open an HTML page in the RealOne Player related info pane, you can use rn:sendTo="_rpcontextwin" in the <area/> link tag:

```
<video src="rtsp://helixserver.example.com/video1.rm">
  <area href="http://www.example.com/info.htm" external="true" alt="More Info"
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpcontextwin"/>
</video>
```

You can set the related info pane's height and width in pixels by using <rn:param/> tags. To do this, turn the <area/> tag into a tag pair (<area> and </area>) that surrounds the <rn:param/> tags:

```

<video src="rtsp://helixserver.example.com/video1.rm">
  <area href="http://www.example.com/info.htm" external="true" alt="Related Info"
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpcontextwin">
    <rn:param name="width" value="300"/>
    <rn:param name="height" value="240"/>
  </area>
</video>

```

For More Information: See “Related Info Pane Sizing” on page 12 for basic information about the related info pane size. “Making Room for the Related Info Pane” on page 81 explains how to open the related info pane when a presentation starts, even when an HTML page does not display in that pane right away.

Hyperlinking Examples

Once you understand the basics of how a SMIL hypertext link works, you can apply some of SMIL’s timing features to open HTML pages at various points as a clip plays. Remember, though, that opening a Web page requires some of the viewer’s bandwidth. If your streaming media uses all of the viewer’s available bandwidth, opening a Web page will be difficult. You may need to ensure that your streaming media uses less bandwidth than the maximum listed in the table “Maximum Streaming Rates” on page 25.

Tip: To get playable examples, download the zipped HTML version of this guide as described in “How to Download This Guide to Your Computer” on page 3, and view the **Sample Files** page.

Opening Several Web Pages During a Presentation

The following markup uses a series of <area/> tags to open three HTML pages in the related info pane at different times as a video plays. Note that only the first link tag sets the related info pane height and width. Because this size is set for the duration of a presentation, subsequent links do not need sizing information. Although all the following links open in the related info pane, you could also add <area/> tags that use rn:sendTo="_rpbrowser" to open HTML pages in the media browser pane as well:

```
<video src="rtsp://helixserver.example.com/video1.rm">
  <!-- First hypertext link, with related info pane sizing information. -->
  <area href="http://www.example.com/info1.html" external="true" begin="10s"
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpcontextwin">
    <rn:param name="width" value="300"/>
    <rn:param name="height" value="240"/>
  </area>
  <!-- Second hypertext link. -->
  <area href="http://www.example.com/info2.html" external="true" begin="1min"
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpcontextwin"/>
  <!-- Third hypertext link. -->
  <area href="http://www.example.com/info3.html" external="true" begin="1.75min"
    actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpcontextwin"/>
</video>
```

The SMIL `begin` attribute creates a timing offset. In the preceding sample, the first link tag uses `begin="10s"` to activate the link 10 seconds after the video starts to play. The second link uses `begin="1min"` to activate the link one minute after the video starts to play. (Note that each `begin` time is relative to the clip, not to the previous link time.) Because all links use `actuate="onLoad"`, their HTML pages open automatically when the `begin` time is reached.

For More Information: See “Using SMIL Timing Attributes” on page 75 for details about the possible timing values for a `begin` attribute.

Opening Pages on a Mouse Click

A link to an HTML page does not have to open automatically. If you leave out the `actuate="onLoad"` attribute, the link opens only when the viewer clicks the clip. In the following example, the video clip defines four timed hyperlinks. The `begin` and `dur` attributes make each link active for one minute at a different point in the presentation. Viewers therefore display different pages depending on when they click the video clip:

```
<video src="rtsp://helixserver.example.com/video1.rm">
  <area href="http://www.example.com/page1.htm" begin="0s" dur="1min"
    external="true" rn:sendTo="_rpbrowser" sourcePlaystate="pause"
    alt="Go to Page 1"/>
  <area href="http://www.example.com/page2.htm" begin="1min" dur="1min"
    external="true" rn:sendTo="_rpbrowser" sourcePlaystate="pause"
    alt="Go to Page 2"/>
  <area href="http://www.example.com/page3.htm" begin="2min" dur="1min"
    external="true" rn:sendTo="_rpbrowser" sourcePlaystate="pause"
```

```

alt="Go to Page 3"/>
<area href="http://www.example.com/page4.htm" begin="3min"
external="true" rn:sendTo="_rpbrowser" sourcePlaystate="pause"
alt="Go to Page 4"/>
</video>

```

For More Information: See “Specifying a Duration” on page 76 for details about the `dur` attribute.

Using Advanced Hyperlinking Features

RealNetworks Production Guide contains additional information about creating hyperlinks. You can download this guide from <http://service.real.com/help/library/encoders.html>. See the guide’s hyperlinking chapter for instructions on carrying out the following functions:

- Link media clips to other media clips. This lets you play a new clip or SMIL presentation in the media playback pane, or in a pop-up media window, when a hyperlink opens.
- Create image maps. Using `<area/>` tags, you can create rectangular, circular, or polygonal hot spot links that overlay portions of a clip.
- Open links on a keystroke. Keystroke access aids user accessibility, and gives all viewers an alternative to clicking a link.
- Set the tabbing order for links. This defines the order in which RealOne Player highlights hyperlinks when the viewer presses **Tab**.
- Adjust a clip’s volume when a link opens. Using this feature, you can boost or reduce a clip’s volume when a hyperlink opens.

SMIL SEQUENCES

Chapter 6 explains the basics of SMIL, and shows how to open HTML pages when a single clip plays. This chapter builds on that foundation to show you how to play a sequence of clips using SMIL. It also introduces you to some of the SMIL timing attributes that you can use for clips and hyperlinks.

Playing Clips in Sequence

As “Writing a Ram File” on page 34 explained, you can play a sequence of clips by listing the clips in order in a Ram file. If you want to use SMIL timing or linking with the clips, though, it’s easier to define the sequence within a SMIL file. So instead of writing a separate SMIL file for each clip, you write one SMIL file that defines all of the features of the sequence. Your Ram file then links to the single SMIL file.

A sequence is the simplest type of SMIL group to create. Just list the clips within `<seq>` and `</seq>` tags in the order that you want them to play. The following example shows the entire SMIL markup required to play three audio clips in sequence:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <body>
    <seq>
      <audio src="rtsp://helixserver.example.com/song1.rm"/>
      <audio src="rtsp://helixserver.example.com/song2.rm"/>
      <audio src="rtsp://helixserver.example.com/song3.rm"/>
    </seq>
  </body>
</smil>
```

In the preceding example, the second clip begins when the first clip finishes, and the third clip begins when the second clip finishes. A sequence can include any number of clips, and the clips can be of any type. You could add a

video clip to the sequence shown above, for example. When using visual clips, though, you need to define a layout. Chapter 8 tackles layouts in general, but the section “Laying Out a Sequence of Videos” on page 72 offers a “universal layout” for any sequence of videos.

When you enclose clips in `<seq>` and `</seq>` tags, RealOne Player treats the sequence as a single presentation. If each clip in the preceding example is two minutes in length, for example, the RealOne Player status bar indicates that the presentation is six minutes long. Because RealOne Player treats the sequence as a single presentation, viewers can use the timeline slider to seek through all the clips, but cannot choose individual clips through the RealOne Player **Play>Next Clip** command.

Creating Sequences Without the `<seq>` Tag

It is not necessary to use `<seq>` and `</seq>` tags to create a clip sequence. Whenever clips are not listed in a group, RealOne Player plays them in sequence. For instance, the following markup, which has no `<seq>` and `</seq>` tags, plays three audio clips in sequence just like the preceding example:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <body>
    <audio src="rtsp://helixserver.example.com/song1.rm"/>
    <audio src="rtsp://helixserver.example.com/song2.rm"/>
    <audio src="rtsp://helixserver.example.com/song3.rm"/>
  </body>
</smil>
```

When you do not use a `<seq>` group, however, RealOne Player treats each clip as a separate presentation. Suppose that each clip in the preceding example lasts two minutes. When the sequence starts, the RealOne Player status bar indicates that the presentation lasts two minutes. When the first clip ends, RealOne Player’s timeline slider resets, the second clip starts, and the status bar indicates another two-minute presentation. This action repeats when the third clip plays. At any point, the viewer can select a different clip with the RealOne Player **Play>Next Clip** command.

Laying Out a Sequence of Videos

When you play a sequence of visual clips such as videos, you need to define a layout. Otherwise, RealOne Player displays the clips in a checkerboard pattern, which is rarely desirable. However, it’s easy to create a simple layout that sets a

size for the media playback pane, then centers each video in the pane. Chapter 8 explains layouts in detail, but the following sample defines a universal layout that works for most video sequences:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <layout>
      <root-layout width="320" height="240"/>
      <region id="video_region"/>
    </layout>
  </head>
  <body>
    <seq>
      <video src="rtsp://helixserver.example.com/video1.rm" region="video_region"
regPoint="center" regAlign="center"/>
      <video src="rtsp://helixserver.example.com/video2.rm" region="video_region"
regPoint="center" regAlign="center"/>
      <video src="rtsp://helixserver.example.com/video3.rm" region="video_region"
regPoint="center" regAlign="center"/>
    </seq>
  </body>
</smil>
```

The preceding sample sets the size of the media playback pane to 320 pixels wide by 240 pixels high. It then creates one playback region that expands to that same size. Note that each clip includes `region="video_region"` to assign it to the playback region, which uses `id="video_region"`. The `regPoint="center"` and `regAlign="center"` attributes place each clip in the center of the region. This layout thereby accommodates any clip up to 320-by-240 pixels, which is as large of a video that you can stream at high quality over low to moderate connection speeds.

For More Information: For more on bit rates and video sizes, see “Video Dimensions” on page 27.

Adding Clip or Group Information

The SMIL attributes listed in the following table let you override information encoded in a clip. They work like the Ram file parameters described in “Overriding Title, Author, and Copyright Information” on page 43. When

using these attributes in the SMIL file, however, do not also use the equivalent Ram file parameters.

Clip and Group Information Attributes

Attribute	Value	Function
author	<i>author_name</i>	Defines the author name.
copyright	<i>copyright_notice</i>	Provides a copyright notice.
title	<i>title_text</i>	Creates a title for the “Now Playing” list.

Each attribute takes a text string for its value:

```
<video src="rtsp://helixserver.example.com/video1.rm"
title="title" author="name" copyright="date"/>
```

When you leave the <seq> tag out, as described in “Creating Sequences Without the <seq> Tag” on page 72, the individual clip titles display in the RealOne Player “Now Playing” list:

```
<body>
  <audio title="This is Clip 1" .../>
  <audio title="This is Clip 2" .../>
</body>
```

When you group clips within a <seq> or <par> tag, the clip titles are ignored. You should therefore add title, author, and copyright information to the group tag, as shown in the following example:

```
<seq title="This is Sequence 1">
  <audio .../>
  <audio .../>
</seq>
```

For More Information: For information about the <par> tag, see “Playing Clips in Parallel” on page 95. The SMIL file can also define overall presentation information in addition to clip or group information, as described in “Presentation Information” on page 61.

Using SMIL Timing Attributes

SMIL includes many timing attributes that you can use in clips or hyperlinks. The following sections describe the most commonly used attributes. The following table lists the values that you can use in SMIL timing attributes.

Timing Values and Examples

Timing Marker	Specifies	Example	Example Value
h	hours	begin="2.5h"	2 hours, 30 minutes
min	minutes	begin="2.75min"	2 minutes, 45 seconds
s	seconds	begin="15.55s"	15 seconds, 550 milliseconds
ms	milliseconds	begin="670.2ms"	670.2 milliseconds

Note: Decimal values are not required. You can express two seconds as "2s" or "2.0s", for example.

Tip: RealOne Player displays a presentation's elapsed time in one-second increments. You can click the time-elapsed field to display time values to 1/10th of a second, however. This can help you decide what timing values to use with a clip.

For More Information: The endsync attribute, which affects timing in a parallel group, is described in the section "Ending a Parallel Group on a Specific Clip" on page 95.

Setting a Begin Time

The section "Opening Several Web Pages During a Presentation" on page 67 introduced you to the begin attribute, which creates a timing delay. Using the begin attribute, you can vary the point at which a hyperlink becomes active, or a clip starts to play back within the presentation timeline:

```
<video src="rtsp://helixserver.example.com/video1.rm" begin="20.5s"/>
```

In the preceding markup, the begin attribute delays the clip playback for 20.5 seconds. Were this clip in a sequence, there would be 20.5 seconds of blank time before the clip starts.

Tip: You can also use begin in <seq> and <par> tags to delay playback of the entire sequence or parallel group. Parallel groups are described in "Playing Clips in Parallel" on page 95.

Using Internal Clip Begin and End Times

The clipBegin and clipEnd attributes specify a clip's internal timing marks where playback begins and ends. They allow you to play just part of a clip that has an internal timeline, such as an audio, video, or animation clip. They have no effect on groups or static clips such as still images, though. The following example uses clipBegin and clipEnd with a video clip:

```
<video src="video1.rm" clipBegin="10s" clipEnd="50s"/>
```

Here, the clip starts playing at its internal 10-second mark rather than at its encoded beginning. It stops when it reaches its 50-second mark, having played for a total of 40 seconds.

Tip: The clipBegin and clipEnd attributes affect a clip in a SMIL presentation the way that start and end affect a clip listed in a Ram file. For more on start and end, see “Controlling How a Presentation Initially Displays” on page 40.

Warning! Do not use clipBegin and clipEnd with live broadcasts, or when delivering clips with a Web server. These attributes function only for prerecorded clips streamed by Helix Universal Server.

Specifying a Duration

The dur attribute controls how long a clip or group plays. The following example ends the video after 85 seconds, regardless of the length of the clip's internal timeline. If the video's timeline is shorter than 85 seconds, the video's last frame appears frozen onscreen until the duration elapses:

```
<video src="video1.rm" dur="85s"/>
```

A common use of dur is to control how long a static clip such as an image appears onscreen. The following example displays an image for two minutes:

```

```

For More Information: See also the fill attribute, which is described in “Setting Clip Fills” on page 96.

Using Advanced Clip and Timing Features

The clip and timing attributes explained in this chapter are a small sample of the extensive SMIL 2.0 features described in *RealNetworks Production Guide*. You

can download that guide from <http://service.real.com/help/library/encoders.html> to learn how to do the following:

- Define brush objects, which are colored blocks created through SMIL. Brush objects are a handy way to create colored areas in a presentation without using GIF images, for example.
- List sequential clips in a Ram file or even another SMIL file. You can then use a primary SMIL file to create an online application, and easily change the streamed content by modifying the Ram or SMIL file used to give the clip URLs.
- Define a base URL if all of your clips reside together. This makes writing clip source URLs easier when you have a lot of clips.
- Adjust clip transparency to make a clip or its background color fully or partially transparent. You can even select a range of clip colors to turn transparent. These features, part of RealNetworks' extensions to SMIL, let you easily modify transparency values in clips without changing the clips themselves.
- Add text to a SMIL presentation through RealText, a plain text file (.txt), or inline text, which you write directly in the SMIL file. RealText lets you set timing values on text blocks to coordinate them with other clips. A plain text file is handy for simple text needs, and inline text lets you easily annotate your SMIL file to provide onscreen clip titles, for instance.
- Prefetch clip data before a clip plays. Prefetching is a powerful feature that gives you a lot of control over bandwidth management so that you can minimize rebuffering.
- Display transition effects such as fades or wipes when clips start or stop. With over 100 transition effects available, you can add a lot of visual appeal to your presentations.
- Use the end attribute in addition to, or in place of, the dur attribute. The end attribute gives you added flexibility for defining clip durations, and is useful with interactive presentations.
- Repeat clips and groups a specific number of times, or for a specific amount of time, by using the repeatCount or repeatDur attribute.
- Set the minimum and maximum times for clip playback with the min and max attributes. These additional timing capabilities can be useful in any cases.

- Utilize interactive timing attributes to start or stop a clip on an event, such as when the viewer clicks another clip. By using interactive timing, you can create presentations that respond to viewer activity.

SMIL LAYOUT

When you stream more than one clip, you use SMIL to lay out the presentation. The layout defines where each clip appears in the RealOne Player media playback pane. Clips might appear side-by-side, for example, or stacked on top of each other. This chapter introduces you to SMIL's basic layout features. Before you create a layout, be sure to read "SMIL File Basics" on page 60.

Tip: If your presentation displays one clip or a sequence of clips, you may not need to create a layout. Without defining a layout, you can stream a sequence of clips as described in Chapter 7, and create HTML page hyperlinks as explained in Chapter 6.

Setting the Media Pane Size

When you define a layout, you first create one (and only one) *root-layout* area, which sets the size of the RealOne Player media playback pane when the presentation starts. Because this size stays constant throughout the presentation, be sure that the root-layout area is large enough to encompass all of the clips that you plan to play at any point during the presentation.

Considerations For Setting a Root-Layout Size

Calculate the root-layout size based on the sizes of clips that play together, as well as any borders you want to add. When you select your root-layout size, keep the following in mind:

- RealOne Player menus and controls will appear around the media playback pane. If you define a very large root-layout area, some parts of the media playback pane, or some RealOne Player controls, may not appear on the viewer's screen. The smallest computer screen in general use is 640 pixels wide by 480 pixels high.

- If you plan to display HTML pages in the related info pane, your total presentation width is that of the root-layout area, *plus* the related info pane.
- As described in “Controlling How a Presentation Initially Displays” on page 40, you can make the presentation display at double-size or full-screen mode when it starts up. Doing this may affect how you define the root-layout area. For example, most computer screens have a width-to-height ratio of 4:3. Therefore, a root-layout area that also has a 4:3 ratio will scale best in full-screen mode.
- It may help to sketch the layout on paper or with illustration software. In your sketch, position the regions and clips, noting their sizes and the thickness of any borders that should appear around them.

Suppose that you plan to display two clips, one 100 pixels wide and the other 200 pixels wide, side-by-side. If you want a five-pixel border around the clips, for example, the root-layout area needs to be 315 pixels wide:

- 5 pixels from the left edge of the root-layout area to the first clip.
- 100 pixels for the first clip.
- 5 pixels from the right edge of the first clip to the left edge of the second clip.
- 200 pixels for the second clip.
- 5 pixels from the right edge of the second clip to the right edge of the root-layout area.

Defining the Root-Layout

The `<root-layout/>` tag falls between `<layout>` and `</layout>` tags within the SMIL header section. The following example adds the layout section after the `<meta/>` tags:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <meta name="title" content="My First SMIL File"/>
    <meta name="author" content="Pat Morales"/>
    <meta name="copyright" content="(c)2001 Spectacular Media Limited"/>
  <layout>
    <root-layout width="320" height="240"/>
    ...regions defined after the root-layout area...
```



```

    </layout>
  </head>
  <body>
    ...clips defined here...
  </body>
</smil>

```

The `<root-layout/>` tag requires height and width values in pixels. Because clips cannot play in the root-layout area, you need to define at least one playback region in addition to the root-layout area.

For More Information: See “Creating Playback Regions” on page 81. “Adding Background Colors” on page 88 tells how to change the color of the root-layout area.

Making Room for the Related Info Pane

When the media browser pane is attached, a SMIL presentation that plays without an HTML page for the related info pane appears centered above the media browser pane. (For an illustration of this, see “Media Playback Pane Sizing” on page 9.) If an HTML page later opens in the related info pane, the SMIL presentation jumps to the left. To prevent this effect, which can be jarring for the viewer, include `rn:contextWindow="openAtStart"` in the `<root-layout/>` tag:

```
<root-layout width="320" height="240" rn:contextWindow="openAtStart"/>
```

When you use this attribute, the SMIL presentation appears at the left side of the top two panes. Any HTML pages then sent to the related info pane appear at the right side. To prevent height resizing when an HTML page appears, specify the same height for the related info pane that you use in the `<root-layout/>` area. The `rn:contextWindow` attribute has no visible effect when the media browser pane is detached.

Creating Playback Regions

Every visual clip must be assigned to a region, which is a rectangular area that lays within the root-layout area. You don’t necessarily have to create a separate region for each clip. When you play a sequence of clips, for example, you can assign each clip to the same region. When multiple clips play in parallel, though, define a separate region for each clip. Although similar to HTML

frames, SMIL regions can overlap, letting you play a clip in one region in front of a background image in another region, for example.

SMIL Region Possibilities



Adding <region/> Tags

Just below the <root-layout/> tag, you define each region with a <region/> tag. Each <region/> tag must have a unique, user-defined ID. The following sample creates two regions that are the same size as the root-layout area. When regions overlap, the region defined first in the SMIL mark-up appears behind the region defined second, so the background region would appear behind the video region:

```
<layout>
  <root-layout width="320" height="240"/>
  <region id="background_region"/>
  <region id="video_region"/>
</layout>
```

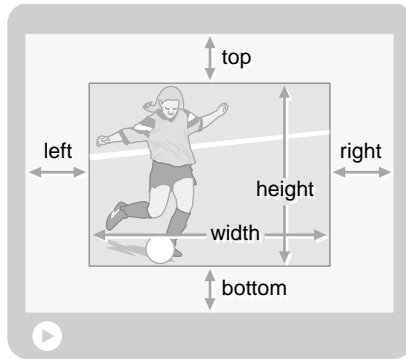
In the preceding sample, you could display an image clip in the background region. Then, if the video is smaller than 320-by-240 pixels, you could center it within the video region. Because region backgrounds are transparent unless you specify otherwise, the background image will display around the video.

For More Information: For more on ID values, see “Tag ID Values” on page 116. “Adding Background Colors” on page 88 explains how to change region colors.

Defining Region Sizes and Positions

In many cases, you'll need to specify region sizes to make them smaller than the root-layout area. This lets you play two clips side-by-side, for instance, or use the root-layout background color as a border around regions. The following figure illustrates how a region's size and position attributes control where the region appears.

Region Size and Position Attributes

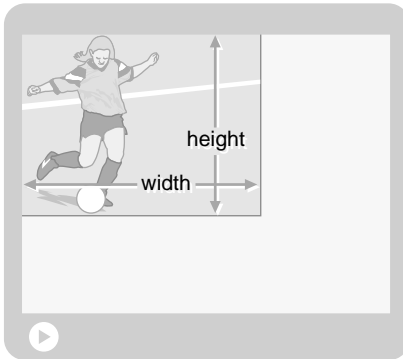


The region size and position attributes constitute a simple coordinate system measured in pixels or percentages. Because each attribute has a default value of auto, you can leave it out of the <region/> tag to set its value automatically based on the values of the other attributes. The result is that, in most cases, you need to specify just one to four of the attributes listed in the following table.

Region Size and Position Attributes

Attribute	Function	Example
bottom	Sets the region's bottom offset from pane's bottom border.	bottom="22"
height	Specifies the region's height.	height="180"
left	Sets the region's left offset from pane's left border.	left="20%"
right	Sets the region's right offset from pane's right border.	right="5%"
top	Sets the region's top offset from pane's top border.	top="60"
width	Specifies the region's width.	width="240"

Layout Example 1: Region Width and Height



This example shows a region in which only the width and height are defined:

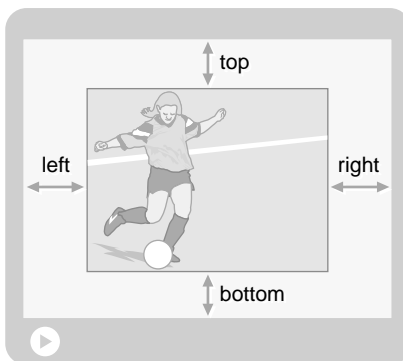
```
<region id="video_region" width="180" height="120"/>
```

In this case, the region is placed at the root-layout area's upper-left corner. The bottom and right offsets from the root-layout borders are set automatically based on the region's size and position. If the root-layout area were 300 pixels wide by 200 pixels high, you could achieve the same layout using percentage values:

```
<region id="video_region" width="60%" height="60%" />
```

Tip: With percentage values, the region changes size if you modify the size of the <root-layout/> tag. With pixel measurements, though, the region size remains stable.

Layout Example 2: Four Region Offsets



This example shows a region placed in a root-layout area without specifying the region size:

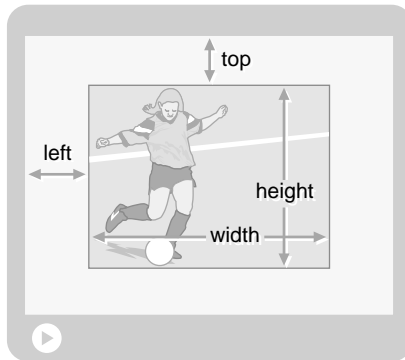
```
<region id="video_region" left="60" right="60" top="40" bottom="40" />
```

In this case, the four offsets from the root-layout borders determine the region size. If the root-layout area were 300 pixels wide by 200 pixels high, the region would be 180 pixels wide ($300-60-60=180$) and 120 pixels high ($200-40-40=120$). You could create the same layout with percentage values:

```
<region id="video_region" left="20%" right="20%" top="20%" bottom="20%" />
```

Tip: If you define a region size with these offset attributes, changing the root-layout area's size also changes the region's size whether the attributes use pixels or percentages.

Layout Example 3: Region Sizes and Two Offsets



This example shows a common way to define region size and position. It specifies a region width and height, then sets the region's offset from the root-layout area's upper-left corner:

```
<region id="video_region" left="60" top="40" width="180" height="120" />
```

If the root-layout area were 300 pixels wide by 200 pixels high, the region layout would be the same as in “Layout Example 2: Four Region Offsets” on page 84. Using pixel measurements for the region width and height, however, keeps the region size stable if you modify the root-layout size.

Using Different Offset Values

For this example, you could use the right and bottom attributes instead of left and top to create the same layout:

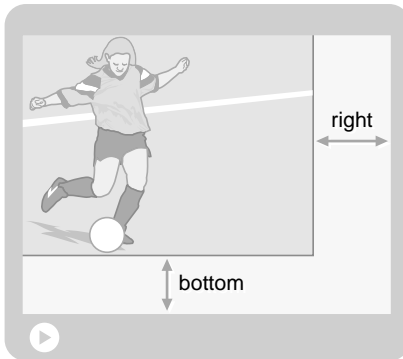
```
<region id="video_region" right="60" bottom="40" width="180" height="120" />
```

Using Percentage Values

You could also define this layout using percentage values for the left and top offsets. This keeps the region's relative position within the root-layout area the same should you change the root-layout size:

```
<region id="video_region" left="20%" top="20%" width="180" height="120"/>
```

Layout Example 4: Two Offsets



This example sets the region's size and position by specifying only the right and bottom attributes:

```
<region id="video_region" right="60" bottom="40"/>
```

Because neither the left nor the top attribute is defined, the region is placed in the root-layout area's upper-left corner. The region's width and height expand to meet the right and bottom offset values.

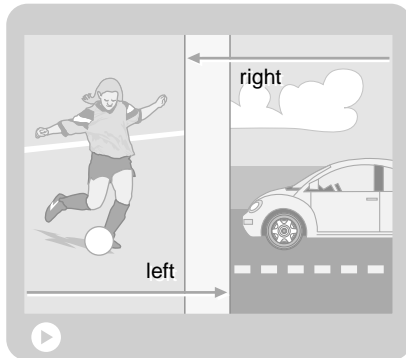
Using Different Offset Values

Alternatively, you could set the region's left and top attributes instead of right and bottom to place the region at the root-layout area's lower-right corner:

```
<region id="video_region" left="60" top="40"/>
```

Layout Example 5: Single Offsets for Two Regions

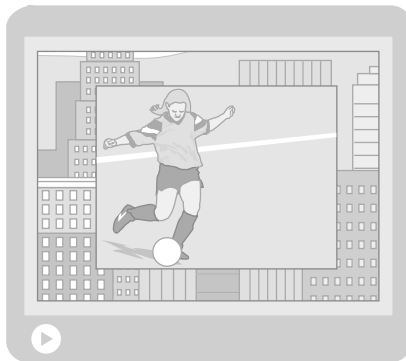
Typically, you'll need to create more than one region to lay out clips that play together. To do this, you define each region with a separate `<region/>` tag, using any combination of size and position attributes to place each region within the root-layout area.



This example shows two regions laid out so that a small stripe of the root-layout background appears between the regions. Because vertical size or offset values (top, height, or bottom) are not specified, each region is as tall as the root-layout area:

```
<region id="region_1" right="55%"/>
<region id="region_2" left="55%"/>
```

Layout Example 6: Overlapping Regions



This example has one region in front of another, the region defined second appearing in front. There are many ways to define this layout with the size and position attributes. The following sample uses percentage values for the four border offsets:

```
<region id="region_1" top="5%" left="5%" bottom="5%" right="5%"/>
<region id="region_2" top="25%" left="25%" bottom="25%" right="25%"/>
```

Tips for Defining Region Sizes and Offsets

- All regions appear within the <root-layout/> area that contains them. Any part of a region defined to appear outside of the root-layout area is cut off. For this reason, no percentage value can effectively be more than 100%.
- You can mix pixel and percentage values. You could define the top and left attributes in percentages, for example, while specifying width and height in pixels.
- You can use both whole and decimal values for percentages. For example, the values “4%” and “4.5%” are both valid.
- An audio clip does not require a region for playback.

Adding Background Colors

By default, the <root-layout/> area has a black background. All regions are transparent. In a <root-layout/> or <region/> tag, you can specify a different background color with the backgroundColor attribute, as shown in the following example:

```
<layout>  
  <root-layout backgroundColor="maroon".../>  
  <region id="region1" backgroundColor="rgb(100,65,230)".../>  
  <region id="region2" backgroundColor="#C2EBD7".../>  
  <region id="region3" backgroundColor="inherit".../>  
</layout>
```

For the color value, you can use inherit to make the region use the same color as the root-layout area. In the example above, the third region inherits maroon as its background color. To set a color value explicitly, use a predefined color name, a hexadecimal color value, or an RGB value. The following table lists the predefined names, along with that color’s corresponding hexadecimal and RGB values.

white #FFFFFF rgb(255,255,255)	silver #C0C0C0 rgb(192,192,192)	gray #808080 rgb(128,128,128)	black #000000 rgb(0,0,0)
yellow #FFFF00 rgb(255,255,0)	fuchsia #FF00FF rgb(255,0,255)	red #FF0000 rgb(255,0,0)	maroon #800000 rgb(128,0,0)

lime #00FF00 rgb(0,255,0)	olive #808000 rgb(128,128,0)	green #008000 rgb(0,128,0)	purple #800080 rgb(128,0,128)
aqua #00FFFF rgb(0,255,255)	teal #008080 rgb(0,128,128)	blue #0000FF rgb(0,0,255)	navy #000080 rgb(0,0,128)

Setting When Background Colors Appear

By default, all region background colors display when the presentation starts. In some cases, you may not want a region's background color to appear until a clip plays in the region. To do this, add `showBackground="whenActive"` to the `<region/>` tag:

```
<region id="region1" backgroundColor="silver" showBackground="whenActive" .../>
```

Transparency in Regions and Clips

If a clip that contains transparency (such as a GIF image) plays in a transparent region, viewers will be able see through the clip's transparent areas to underlying regions and clips. The following clip types can include transparent areas:

- RealVideo
- RealPix
- RealText
- Flash
- GIF and PNG images

Helix Universal Server can stream other types of clips, too, and some of those clips may include transparency. Support for transparency for each clip type has to be built into RealOne Player, however. Some clips that display transparency when rendered in a Web browser, for example, may not display transparency when played in RealOne Player.

Tip: To check if RealOne Player recognizes a clip's transparency, open the clip in RealOne Player and see if the pane background shows through the clip's transparent areas.

Positioning Clips in Regions

By default, a clip aligns with a region's upper-left corner and displays at its normal size. If it's too big for the region, it's cropped. If it's too small, the region's background color displays in the remainder of the region. You can modify this behavior by defining registration points in clip source tags, and adding a fit attribute to <region/> tags.

Creating Registration Points

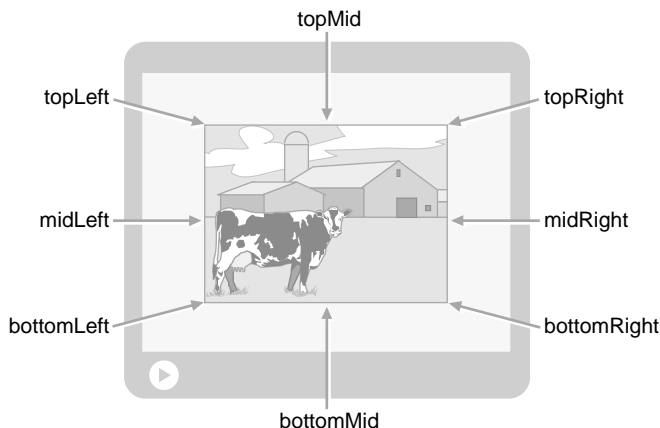
A registration point positions a clip within a region at a point other than the region's upper-left corner. To define a registration point, you add `regPoint` and `regAlign` attributes to a clip source tag. For example, the following values center the clip in its region:

```
<video src="rtsp://helixserver.example.com/video1.rm" region="video_region"
regPoint="center" regAlign="center"/>
```

For More Information: The section “Assigning Clips to Regions” on page 95 explains how the region attribute assigns a clip to a playback region.

Both the `regPoint` and `regAlign` attributes use one of nine values to align a clip within a region: `topLeft`, `topMid`, `topRight`, `midLeft`, `center`, `midRight`, `bottomLeft`, `bottomMid`, or `bottomRight`. The following figure illustrates where these values fall on a clip:

Alignment Values on Clips



Although `regAlign` and `regPoint` both use the same alignment values, the values have different meanings for the two attributes:

- The alignment value used with the `regPoint` attribute determines where the registration point falls in the region (hence, the alignment value applies to the region, not to the clip).
- The alignment value used with the `regAlign` attribute specifies which part of the clip aligns to the registration point.

For example, the next values select the region's lower-right corner, and place the clip's right midpoint at that corner. In this case, the clip's bottom half is cut off:

```
<ref src="rtsp://helixserver.example.com/video1.rm"
regPoint="bottomRight" regAlign="midRight"/>
```

Avoiding Problems When Defining Registration Points

Because you can use any of the nine predefined values for both `regPoint` and `regAlign`, there are 81 possible ways to place clips in regions. Not all possibilities are useful, though. Consider this alignment:





```
<ref src="..." region="..." regPoint="topLeft" regAlign="bottomRight"/>
```

In the preceding example, `regPoint="topLeft"` puts the registration point at the region's upper-left corner. The `regAlign="bottomRight"` attribute places the clip's lower-right corner on the registration point. This locates the clip outside the region. Because a clip cannot display outside its region, the clip does not display at all.

Using Common Registration Point Values in Clip Source Tags










The following table lists some of the more useful combinations of `regPoint` and `regAlign` that you can include in a clip source tag.

Common Registration Point Values in Clip Source Tags

Clip Placement	Registration Point Values	Example
top left (default)	<code>regPoint="topLeft" regAlign="topLeft"</code>	
top center	<code>regPoint="topMid" regAlign="topMid"</code>	
top right	<code>regPoint="topRight" regAlign="topRight"</code>	
middle left	<code>regPoint="midLeft" regAlign="midLeft"</code>	

(Table Page 1 of 2)

Common Registration Point Values in Clip Source Tags (continued)

Clip Placement	Registration Point Values	Example
center	regPoint="center" regAlign="center"	
middle right	regPoint="midRight" regAlign="midRight"	
bottom left	regPoint="bottomLeft" regAlign="bottomLeft"	
bottom center	regPoint="bottomMid" regAlign="bottomMid"	
bottom right	regPoint="bottomRight" regAlign="bottomRight"	
upper-left quadrant	regPoint="center" regAlign="bottomRight"	
upper-right quadrant	regPoint="center" regAlign="bottomLeft"	
lower-left quadrant	regPoint="center" regAlign="topRight"	
lower-right quadrant	regPoint="center" regAlign="topLeft"	

(Table Page 2 of 2)

Defining How Clips Fit Regions

Whereas a registration point determines where a clip displays in a region, a fit attribute specifies what happens when a clip is larger or smaller than its allotted area. The various fit values determine whether resizing, distortion, and cropping may occur. The fit attribute is part of a <region/> tag, and it applies to a clip played in the region whether or not the clip uses a registration point. The following example shows a fit attribute set in a <region/> tag:

```
<region id="video_region" width="128" height="64" fit="meet"/>
```

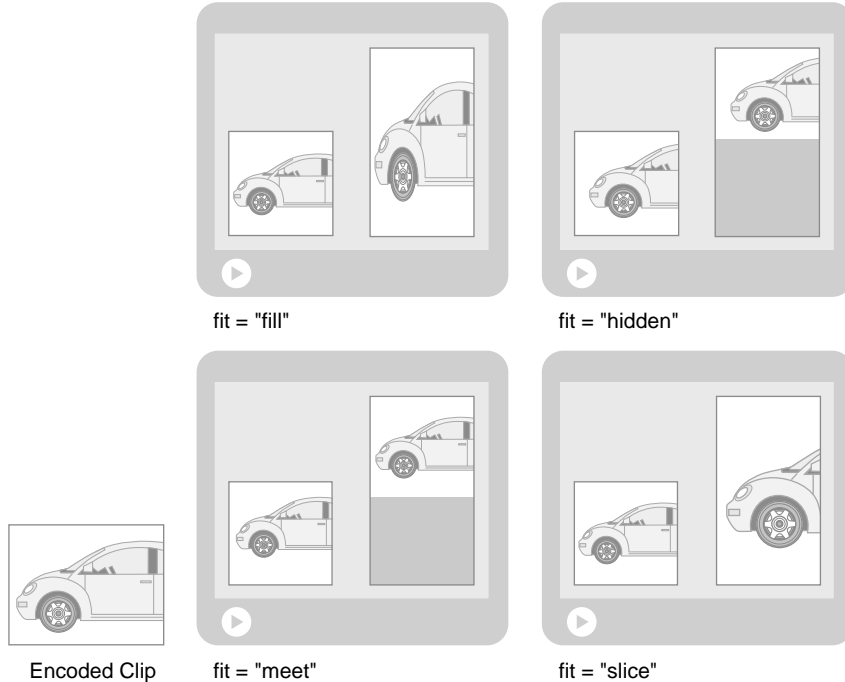
Using fit Attribute Values

The fit attribute uses one of the values described in the following table. The table's last three columns indicate if the fit attribute value may scale, distort, or crop the clip if it does not fit the region dimensions exactly.

fit Value	Function	Scaling?	Distortion?	Cropping?
fill	Place the clip in the region's upper-left corner, or at the registration point. Scale the clip so that it fills the region exactly. Image distortion occurs if the encoded clip and playback region have different aspect ratios.	yes	yes	no
hidden (default)	Keep the clip at its encoded size, and place it in the region's upper-left corner, or at the registration point. If the clip is smaller than the region, fill the remaining space with the region's background color. If the clip is larger than the region, crop out the area that does not fit.	no	no	yes
meet	Place the clip at the region's upper-left corner or at the registration point. Scale the clip and preserve its width-to-height ratio until one dimension is equal to the region's size and the other dimension is within the region's boundaries. Fill empty space with the region's background color.	yes	no	no
scroll	Place the clip at the region's upper-left corner or at the registration point. Display the clip as its normal size, adding horizontal or vertical scroll bars if the clip extends beyond the region's boundaries.	no	no	no
slice	Place the clip at the region's upper-left corner or at the registration point. Scale the clip and preserve its width-to-height ratio until one dimension is equal to the region's size and the other dimension overflows the region's boundaries. Crop the overflow.	yes	no	yes

The following illustration shows the effects that particular fit attribute values have on a source clip played in regions with different sizes and aspect ratios. Note that in some cases, based on the width-to-height ratio of the clip and the width-to-height ratio of the region, certain fit values have nearly the same effect. But display the same clip in a region with a different width-to-height ratio, and the fit values can have very different effects.

A Clip Played in Different Regions with Different fit Attribute Values



Tips for Defining the fit Attribute

- Use fit="meet" if all parts of the clip must display, if the clip's aspect ratio must be maintained, and if it's OK to scale the clip.
- Use fit="hidden" or fit="scroll" to keep the clip at its encoded size.
- Use fit="fill" if you want to fill the entire region with the clip and it doesn't matter if RealOne Player enlarges, shrinks, or distorts the clip.
- When scaling clips inside a region, keep in mind that different types of media scale with different results. A video scaled to a different width-to-height ratio may not look good. Vector-based media such as Flash animation, on the other hand, scale more easily to fit different region sizes. Also, note that scaling a clip consumes CPU power on the RealOne Player computer.

Assigning Clips to Regions

After you define the playback regions, you use region attributes within clip source tags to assign visual clips to regions based on the region's ID. (Audio clips do not require playback regions.) In the following example, the video and text clips are assigned to the video and text regions defined in the header. This sample places clips in a parallel group, which the next section describes:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <layout>
      <root-layout backgroundColor="maroon" width="250" height="230"/>
      <region id="video_region" top="5" left="5" width="240" height="180"/>
      <region id="text_region" top="200" left="5" width="240" height="20"/>
    </layout>
  </head>
  <body>
    <par>
      <video src="video.rm" region="video_region" .../>
      <audio src="audio.rm"/>
      <ref src="text.rt" region="text_region" .../>
    </par>
  </body>
</smil>
```

Playing Clips in Parallel

You can play two or more clips at the same time by grouping the clip source tags between `<par>` and `</par>` tags. In this case, you assign each clip to a different playback region. The following example creates a parallel group that plays a RealVideo clip along with a RealText clip in separate regions at the same time. The group ends when the longer clip in the group stops playing:

```
<par>
  <video src="song.rm" region="video_region"/>
  <ref src="lyrics.rt" region="text_region"/>
</par>
```

Ending a Parallel Group on a Specific Clip

By default, a `<par>` group ends when all clips in the group finish playing. You can modify this behavior with the `endsync` attribute. Suppose that a long clip of background music plays in parallel with a shorter RealText clip. Using

endsync, you can stop the group when the RealText clip finishes, cutting off the background music once the text has displayed. The endsync attribute has no effect in <seq> tags or clip source tags.

Use endsync="first" in the <par> tag to stop the group when the first clip in the group stops playing. (Note that "first" refers to playback times and not the order that clips are listed in the group.) All other clips in the group stop playing at that point, regardless of their playback statuses or any timing parameters specified for them.

The attribute endsync="ID" causes the group to conclude when the designated clip ends playback. All other clips in the group stop playing at that point, regardless of their playback statuses or any timing parameters used with them. The designated clip must have a corresponding id value in its source tag, as illustrated in the following example:

```
<par endsync="vid1">  
  <video id="vid1" src="video1.rm" region="video_region"/>  
  <ref src="moreinfo.rt" region="text_region"/>  
</par>
```

For More Information: For more on clip ID values, see "Tag ID Values" on page 116.

Setting Clip Fills

The fill attribute, which uses three basic values of auto, remove, and freeze, is useful with clips that play in parallel groups. If a group lasts longer than a clip, use fill="remove" in the clip source tag to remove the clip when it finishes playing. A fill="freeze" value keeps the clip visible until the group completes.

In the following example, suppose that the video is longer than the RealText clip. The fill="freeze" attribute in the RealText source tag keeps the final text block visible after the clip finishes playing. The parallel group then ends with the video clip finishes playing:

```
<par>  
  <video src="song.rm" region="video_region"/>  
  <ref src="lyrics.rt" region="text_region" fill="freeze"/>  
</par>
```

If you do not set the fill value explicitly, the default value of fill="auto" is used. This acts like a remove or a freeze value depending on whether timing attributes are used:

- If the clip includes a `dur` attribute, the `fill="auto"` value is equivalent to `fill="remove"`. For example, a video that uses a `dur` attribute disappears when the duration expires.
- If the clip does not include a `dur` attribute, the `fill="auto"` value is equivalent to `fill="freeze"`. For example, the final frame of a video that does not use a `dur` attribute freezes until the group that contains the clip ends.

Using Images in Parallel Groups

You can stream still images, whether GIF, JPEG, or PNG, in parallel with other clips, using the images as buttons, banners, or ads, for example. Still images lack two characteristics that you find in a streaming media clip such as a video, though:

- Still images do not have natural durations. Through the SMIL `dur` attribute, which is described in the section “Specifying a Duration” on page 76, you can define the duration for each image.
- Still images stream at a preset speed of 12 Kilobits per second (12 Kbps). You can change this streaming speed with SMIL, however.

The following example shows a clip source tag for a still image. The `<param/>` tag sets the image’s streaming speed to 5000 bits per second, or approximately 5 Kbps. Note that using `<param/>` requires that you turn the `` tag into a tag pair of `` and ``:

```

  <param name="bitrate" value="5000"/>
</img>
```

Note: You see the effects of the `bitrate` parameter only when streaming an image with Helix Universal Server. The parameter does not affect local playback of clips stored on your computer. When images are on a Web server, the `bitrate` parameter has no effect, and the images simply download as fast as possible, which may interfere with other streaming clips playing in parallel.

Tips for Creating Parallel Groups

- When you create parallel groups, you need to be careful that clips playing at the same time do not exceed the audience connection’s maximum bandwidth, which is described in “Audience Bandwidth Targets” on page

25. If the maximum streaming bandwidth is 34 Kbps, for example, do not have two clips that each stream 20 Kbps of data play in parallel.
- A `<par>` tag can include a title, author, or copyright attribute just like a clip source tag. For more information, see “Adding Clip or Group Information” on page 73.
 - When a parallel group contains a still image, RealOne Player does not play the group until it has received all of the image data. If you set too low of a streaming speed for an image, therefore, you may delay group playback.
 - To determine how many seconds an image will take to stream, take the file size in Kilobytes, multiple by 8192 to get the number of bits, then divide by the streaming speed set through the `<param/>` tag’s bitrate parameter. For example, a 10 Kilobyte file streaming at 20,000 bits per second takes a little over four seconds to stream $((10 \times 8192)/20000)=4.1$.
 - A `dur` in a `<par>` tag overrides `endsync`. In these cases, RealOne Player ends the group as specified by the `dur` attribute, not the `endsync` attribute.
 - You can also use the attribute and value `endsync="last"` in a `<par>` group to end the group when the last clip finishes playing. (Here, “last” refers to playback times and not the order that clips are listed in the group.) Because `endsync="last"` is the group default, however, it is not necessary to add this attribute and value to the `<par>` tag explicitly.
 - Be careful not to confuse `fit="fill"` in a `<region/>` tag, as described in “Defining How Clips Fit Regions” on page 92, and `fill="freeze"` or `fill="remove"` in a clip source tag. In the former case, `fill` is a *value* for the `fit` attribute, which affects clip sizing in regions. In the latter case, `fill` is an *attribute* that affects clip timing and appearance.

Defining Groups Within Groups

One of the powerful features of SMIL is the ability to nest groups, which lets you combine `<seq>` and `<par>` tags as needed to create any type of presentation. The organization of these tags greatly affects the presentation playback, though, and you need to be careful when creating nested groups. In the following example, clip 1 plays first. When it finishes, clip 2 and clip 3 play together. When both clip 2 and clip 3 have finished playing, clip 4 plays:

```

<seq>
  clip 1
  <par>
    clip 2
    clip 3
  </par>
  clip 4
</seq>

```

You get very different results, though, if you switch the `<seq>` and `<par>` groupings. In the next example, clips 1, 2, and 4 all begin at the same time. When clip 2 finishes, clip 3 starts:

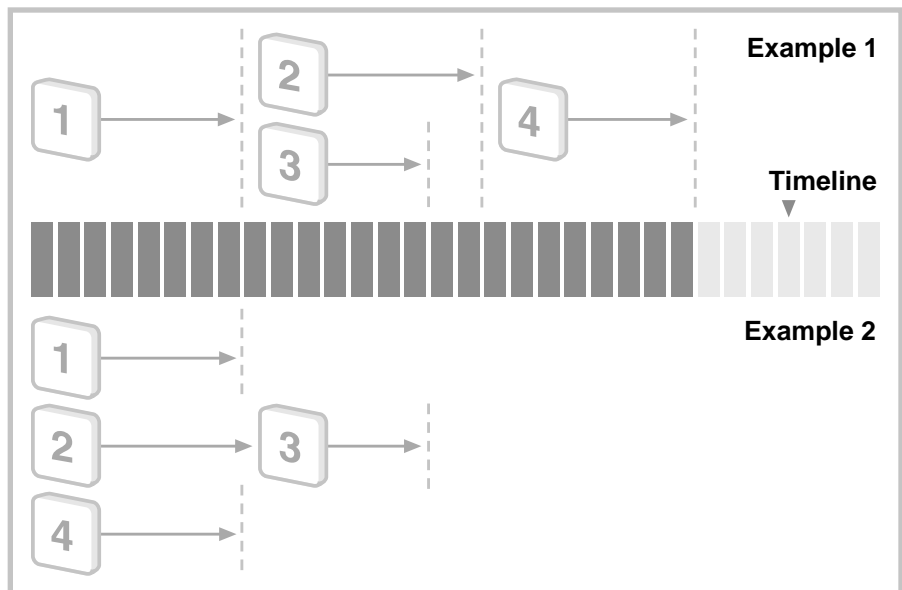
```

<par>
  clip 1
  <seq>
    clip 2
    clip 3
  </seq>
  clip 4
</par>

```

The following illustration shows the difference between these groupings.

Different Playback Results with Nested Groups



Layout Examples

The following sections illustrate how to use layout tags and attributes to create various types of presentations. To get playable examples, download the zipped HTML version of this guide as described in “How to Download This Guide to Your Computer” on page 3, and view the **Sample Files** page.

Centering a Video on a Background

This example centers a video clip in front of an image. Because region sizes are not specified, the regions expand to the root-layout size. The registration point centers the video clip within its region. The image region’s `fit="fill"` attribute expands the image to fill the entire region, distorting the image if the image does not have the same width-to-height ratio as the region:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <layout>
      <root-layout width="320" height="240"/>
      <region id="image_region" fit="fill"/>
      <region id="video_region"/>
    </layout>
  </head>
  <body>
    <par>
      
      <video src="rtsp://helixserver.example.com/video1.rm" region="video_region"
        regPoint="center" regAlign="center"/>
    </par>
  </body>
</smil>
```

Note: SMIL provides no way to tile an image throughout a region.

Displaying a Letterbox Clip

A wide screen movie displays on most television sets in a letterbox format, in which blank areas display above and below the movie. As shown in the following example, you can achieve the same effect for a clip that has a width-to-height ratio greater than its region’s. Here, the video uses a registration

point that centers it in a region that uses `fit="meet"` to scale the video up or down in size until its left and right edges meet the region boundaries:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <layout>
      <root-layout width="400" height="300"/>
      <region id="video_region" fit="meet"/>
    </layout>
  </head>
  <body>
    <video src="rtsp://helixserver.example.com/widescreen.rm"
      region="video_region" regPoint="center" regAlign="center"/>
  </body>
</smil>
```

Playing Three Clips Side-by-Side

The following example displays three regions: a news region, a video region, and a stock ticker region. The news and video regions are arranged side-by-side at the top of the RealOne Player media playback pane. The stock ticker region appears below them. The video uses a `begin` attribute to start it playing 15 seconds after the other clips have started.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  <head>
    <layout>
      <root-layout height="230" width="510" backgroundColor="black"/>
      <region id="news_region" width="240" height="180" left="5" top="5"/>
      <region id="video_region" width="240" height="180" right="5" top="5"/>
      <region id="ticker_region" width="500" height="30" left="5" bottom="5"/>
    </layout>
  </head>
  <body>
    <par>
      <ref src="rtsp://helixserver.example.com/news.rt" region="news_region"
        fill="freeze"/>
      <video src="rtsp://helixserver.example.com/video1.rm" region="video_region"
        fill="freeze" begin="15s"/>
      <ref src="rtsp://helixserver.example.com/ticker.rt" region="ticker_region"
```

```
        fill="freeze"/>  
    </par>  
</body>  
</smil>
```

Using Advanced Layout Features

SMIL 2.0 has even more layout features than those described in this chapter. If you download *RealNetworks Production Guide* from <http://service.real.com/help/library/encoders.html>, you'll find instructions for doing the following:

- Open media clips in secondary windows that pop up above the media playback pane. This lets you play multiple clips without confining them to a single root-layout area.
- Control the resize behavior of clips if the viewer manually changes the size of the media playback pane.
- Set each region's z-index attribute to define explicitly which region appears in front when regions overlap. This way, you do not need to rely on the order in which regions are defined within the SMIL file.
- Adjust a clip's audio level as it plays in a region. This lets you boost or cut a clip's volume automatically.
- Define subregions that are associated with larger playback regions. Using a subregion, you can easily display a semi-transparent GIF logo on top of a video, for example.
- Create a registration point in the SMIL header and apply it to any number of clips and regions. When you have a lot of clips that use the same registration point, this approach is easier than defining the same `regPoint` and `regAlign` values in each clip source tag.

QUICK ANSWERS

This appendix provides quick answers to common questions about producing streaming media clips. It also provides URLs for Web sites where you can find tools and helpful information about developing streaming presentations.

Playing Media with RealOne Player

RealOne Player plays the media clips that you create. It can also display HTML pages that accompany your media presentation. You can download RealOne Player from <http://www.real.com>. See Chapter 1 for an introduction to the RealOne Player interface.

Is a subscription required to view media with RealOne Player?

No. RealOne Player includes a subscription service that provides premium media content and music. But RealOne Player is designed to be a general-purpose media player for any type of free or paid media content.

Must a presentation played in RealOne Player include HTML pages?

No. RealOne Player can display HTML pages along with media, a combination that greatly enhances the viewing experience. You can also stream media alone, though, without displaying HTML pages along with your clips.

What HTML page technologies does RealOne Player support?

On Microsoft Windows, RealOne Player uses the existing version of Internet Explorer. Because Internet Explorer 4 is the earliest version that functions with RealOne Player, writing HTML content that can play in this browser guarantees access to the widest possible audience. This supported set of technologies includes Javascript 1.2 and Cascading Style Sheets 1 (CSS1).

Can I embed streaming media directly in a Web page?

Yes. You can still use RealOne Player to embed media clips directly into any Web page, as described in the Web page embedding chapter of *RealNetworks Production Guide*. However, the native RealOne Player interface provides an easier way to coordinate media and HTML pages, eliminating the cumbersome markup required to embed a presentation.

How can I protect copyrights on media?

The extensive Media Commerce Suite allows you to protect copyrights for valuable media assets. You can learn more about this suite from the following Web page:

<http://www.realnetworks.com/products/commerce/index.html>

Creating Streaming Clips

Helix Producer is the basic tool you use to create clips. Both the *Helix Producer User's Guide* and the product's online help guide you through the encoding process.

How do I make streaming audio and video clips?

You start with an audio or video source file in a digitized format on your computer. You then use Helix Producer to select the file and set encoding options. The encoding process creates a new streaming clip, leaving the source file unchanged.

Can I encode RealVideo directly from a video camera?

Yes. Helix Producer accepts live video input from a camera and live audio input from a microphone. The camera and microphone connect to an audio/video capture card on your computer. Helix Producer then lets you select the live input as the source. In this case, you go directly from live input to encoded clip without creating a digitized source file.

How do I ensure the best quality for streaming clips?

Quality starts at the source. You need high-quality video and audio input for Helix Producer to create high-quality streaming clips. If you are new to media production, learn your editing hardware and software thoroughly, paying close attention to the manufacturers' recommendations for producing high-

quality media files. See also the tips in *Helix Producer User's Guide*, as well as the audio chapter and video chapter of *RealNetworks Production Guide*.

What other clips can I stream?

In addition to audio and video, Helix Universal Server can stream the following types of clips:

- Macromedia Flash animation
- GIF, JPEG, and PNG images
- RealPix clips for streaming slideshows
- RealText clips for streaming text

Where do I get a RealOne Player download logo?

When you create streaming media content, you can provide a RealOne Player download link so that users can get RealOne Player from RealNetworks' Web site and view your content. You can read RealNetworks' trademark policies and get RealOne Player logos at the following address:

<http://www.realnetworks.com/company/logos/index.html>

How do I advertise my streaming clips and events?

Every day, thousands of people visit RealGuide, RealNetworks' online guide for streaming media sites and live events (**<http://realguide.real.com>**). If you regularly host streaming media presentations of interest to the public, or if you have a live event you want to advertise, you can submit your listing to RealNetworks. Simply complete the following online form to list your site or live event:

<http://realguide.real.com/info/?page=submit>

Getting Production Tools

To produce streaming media clips, you need audio and video production tools as well as Helix Producer to handle the encoding.

What audio and video editing tools can I use?

You can use any hardware or software designed for capturing and editing audio or video. The digitized output must be in a format that Helix Producer accepts, however. Some video editing programs save digitized video in a

proprietary format that Helix Producer cannot read. However, these programs typically let you export the video to a common format that Helix Producer accepts, such as AVI, QuickTime, or MPEG.

Tip: Check <http://www.real.com/accessories/index.html> for hardware and software tools that can help you with capturing and editing audio or video.

What digitized audio and video formats does Helix Producer accept as input?

Helix Producer accepts many common audio and video formats. These may vary by operating system, though. Helix Producer on Macintosh accepts the formats widely used on the Macintosh, such as QuickTime, whereas Helix Producer on Windows or Unix supports the formats widely used on those operating systems. Check the Helix Producer manual for your operating system for a list of accepted formats. Information is also available at the following Web page:

<http://www.realnetworks.com/products/producer/features.html>

Where can I get Helix Producer?

RealNetworks makes versions of Helix Producer for Windows, Macintosh, and Linux. You can download the free version or purchase Helix Producer Plus at RealNetworks' Web site:

<http://www.realnetworks.com/products/producer/index.html>

How do I create a streaming slideshow from still images?

Using RealSlideshow's graphical interface, you can create streaming RealPix presentations from still images. You can even add a soundtrack, or record a narration for each image. You can download RealSlideshow from this Web address:

<http://www.realnetworks.com/products/slideshow/index.html>

You can also create RealPix presentations by hand with the RealPix markup language, which is described in *RealNetworks Production Guide*, available at this Web page:

<http://service.real.com/help/library/encoders.html>

How do I create streaming Flash animation?

You create animation with Macromedia Flash. You can develop animations with Flash 2, 3, or 4. The Flash chapter in *RealNetworks Production Guide* provides tips for making Flash animation stream well with Helix Universal Server. It doesn't explain how to create Flash animations, however. You can learn more about Flash from Macromedia's Web site:

<http://www.macromedia.com/software/flash/>

Using SureStream

SureStream provides advanced streaming technology. For more information about SureStream, read "SureStream RealAudio and RealVideo" on page 25.

What is SureStream?

SureStream is a technology that lets a single RealAudio or RealVideo clip stream at different bit rates. It does this by bundling into a single clip multiple streams, each of which runs at a different bit rate. You can make a SureStream clip that streams at either 28.8 Kbps or 56 Kbps, for example. When users request the clip, they automatically receive the stream that best matches their RealOne Player connection speed.

How do I make a SureStream clip?

Using Helix Producer, you can choose to use SureStream when you encode audio or video input. The number of SureStream streams you can encode in the clip depends on the type of Helix Producer you use. Helix Producer Basic encodes three speeds per clip, whereas Helix Producer Plus encodes an unlimited number of speeds per clip.

Can I use SureStream with a Web server?

No. A SureStream clip has several streams encoded in a single clip. Unlike Helix Universal Server, a Web server cannot extract a specific stream to send to RealOne Player. If you plan to deliver clips from a Web server, you need to set Helix Producer to use single-rate encoding.

Writing SMIL Files

Chapter 6 explains the basics of SMIL. Appendix B explains important concepts and rules to follow when writing SMIL.

What is SMIL?

Pronounced “smile,” SMIL stands for “Synchronized Multimedia Integration Language.” It is an industry-standard markup language used to lay out and time streaming media presentations. SMIL works for RealOne Player the way HTML works for a Web browser.

Is it necessary to use SMIL?

Not always. When you want to stream just one clip, such as a single video clip, you don’t need to use SMIL. You just link your Web page to the clip through a Ram file, as described in Chapter 3.

When should I use SMIL?

When you stream multiple clips, SMIL gives you the means to lay out the presentation and time its clips. It also provides other features, such as letting you create hyperlinks that display HTML pages, or that start new media presentations.

How do I write SMIL?

SMIL is a simple markup language that you can write with a word processor or text editor. Some software tools (RealSlideshow, for example) create SMIL files automatically. Other SMIL editing tools are also available. Visit the following Web page for more information:

http://www.realnworks.com/products/media_creation.html

What’s the difference between SMIL 1.0 and SMIL 2.0?

As the numbers suggest SMIL 2.0 is an enhancement to SMIL 1.0, which was introduced in 1998. SMIL 2.0 greatly expands the capabilities of SMIL 1.0. Because it is newer than SMIL 1.0, though, not every media player that supports SMIL 1.0 can handle SMIL 2.0. RealOne Player can handle both SMIL 2.0 and SMIL 1.0. RealPlayer G2, RealPlayer 7, and RealPlayer 8 can read only SMIL 1.0 files, however.

What advanced SMIL features can I use?

If you are intrigued by SMIL, download a copy of *RealNetworks Production Guide* from **<http://service.real.com/help/library/encoders.html>**. In that reference manual, you’ll find more information about the SMIL features covered in this guide, as well as instructions for performing tasks such as the following:

- Modify image and background transparencies to make an image semi-transparent, for example. This is a great way to superimpose a logo over a video.
- Open media clips in secondary windows that pop up above the media playback pane.
- Link media clips to other media presentations.
- Overlay media clips with clickable image maps.
- Display different clips or HTML pages based on the viewer's preferred language, available bandwidth, monitor color depth, or many other criteria.
- Make your presentation more accessible to all viewers through captions or audio descriptions that play for the viewers requesting them.
- Display special effects such as fades when a clip starts or stops playing.
- Dynamically change clip sizes or positions during a presentation.
- Manage bandwidth for complex presentations by prefetching clip or HTML page data before the clips or pages display.
- Boost or reduce a clip's audio volume as it plays in a region.
- Build interactive applications that respond when a viewer clicks a clip.

Streaming Clips

Helix Universal Server streams the clips created by Helix Producer. You can stream clips yourself with Helix Universal Server, through a service provider that has Helix Universal Server available, or, in some cases, from a Web server.

Do I need to install Helix Universal Server on my desktop computer?

Not necessarily. To run Helix Universal Server, you need a computer connected to an intranet or one that has a direct presence on the Internet. You cannot run Helix Universal Server if you use an Internet service provider (ISP) to connect to the Internet. If you use an ISP, check whether they have Helix Universal Server and whether they can host your streaming presentations for you.

What operating systems does Helix Universal Server run on?

Helix Universal Server runs on Windows NT/2000/XP and many Unix platforms, including Linux. For a list of available platforms, visit RealNetworks' technical support Web site at <http://service.real.com>.

Where do I get Helix Universal Server?

Helix Universal Server is available on the RealNetworks Web site at http://www.realnetworks.com/products/media_delivery.html. Helix Universal Server Basic is free.

Can I stream clips from a Web server instead of Helix Universal Server?

Sometimes. A Web server can deliver many types of clips, including RealAudio and RealVideo. There are limits to Web server delivery, however. If you plan to use a Web server for clip delivery, you cannot use SureStream RealAudio and RealVideo clips. Additionally, you cannot use the SMIL `clipBegin` and `clipEnd` attributes described in "Using Internal Clip Begin and End Times" on page 76. *RealNetworks Production Guide* has a section on Web server limitations in its presentation delivery chapter.

What is a Ram file?

A Ram file, also called a *metafile*, is a simple text file with the extension `.ram`. It typically consists of just one line: the URL to a streaming presentation. Your Web page does not link directly to your presentation. Instead, it links to the Ram file, which ensures that RealOne Player launches. RealOne Player then uses the URL in the Ram file to request the presentation. Chapter 3 explains how to write a Ram file.

If I use SMIL, do I need a Ram file?

Yes. The SMIL file lists the URLs for clips. The Ram file supplies RealOne Player with the URL to the SMIL file (or to your streaming clip, if you're not using SMIL). The Ram file is always necessary because its `.ram` extension launches RealOne Player.

Why does Helix Universal Server use RTSP rather than HTTP?

Web servers use HTTP to deliver Web pages and graphics. HTTP is designed to download small files quickly and efficiently. It is not suited for streaming large media clips, though. RTSP, which stands for "RealTime Streaming Protocol," is an industry-standard protocol that overcomes the deficiencies of HTTP for

streaming media. RTSP enables Helix Universal Server and RealOne Player to stream long clips and compensate for changing network conditions.

How do I stream clips with RTSP?

When a clip resides on Helix Universal Server, make sure that the URL used to request it starts with `rtsp://` rather than `http://`. An RTSP URL must be in a file read by RealOne Player, such as a Ram file or a SMIL file. It cannot be in an HTML page hyperlink, because a Web browser does not know how to make an RTSP request. For more on this, see “Why Does Helix Universal Server Use RTSP?” on page 36.

Broadcasting

For full information about broadcasting media, see *Helix Producer User's Guide* and *Helix Universal Server Administration Guide*.

What do I need for broadcasting over a network?

You need the following:

- An audio or video capture card on your computer, to digitize the input from a microphone or camera.
- Helix Producer on the same computer as the capture card, to encode the output in a streaming format and send the stream to Helix Universal Server.
- Helix Universal Server, to broadcast the stream to one or more RealOne Players. Helix Universal Server typically does not run on the same computer as Helix Producer.

Can I broadcast through my ISP?

Possibly. If you connect to the Internet through an ISP, you may be able to broadcast streaming media, provided that your ISP has Helix Universal Server available and offers broadcasting services. To do this, you will need a fast Internet connection to your ISP. You cannot broadcast through an ISP by running Helix Universal Server on your desktop computer.

Can I use SureStream in a broadcast?

Yes. Using SureStream is recommended because it ensures that users connecting at different speeds will each receive the best possible stream. You

need to make sure, however, that the computer running Helix Producer has enough power to encode all the SureStream streams at the same time. Check Helix Producer's manual or online help for system requirements, and perform a trial run before streaming the actual broadcast.

Can I broadcast with a Web server instead of Helix Universal Server?

No. You need Helix Universal Server to broadcast streaming presentations. Web servers are designed to serve HTML pages and graphics to different users at different times. They are not designed to broadcast the same presentation to multiple users simultaneously.

Does a broadcast have to be live?

No. "Broadcasting" means to send out a stream that more than one RealOne Player user can view at the same time. The broadcast can be live, meaning that the input originates from a microphone or video camera. Or it can be prerecorded, meaning that it originates from a digitized clip prepared in advance. If it's prerecorded, you don't need to use Helix Producer during the broadcast. You just put the clip on Helix Universal Server and then set up Helix Universal Server to broadcast the clip as a simulated live event.

Can I use SMIL with a broadcast?

Yes. You can use SMIL to include ads with the broadcast, or deliver static clips alongside the broadcast. In the SMIL file, you simply treat the broadcast as a static clip. The only difference is that you use a special URL created by the Helix Universal Server administrator that identifies the resource as a broadcast rather than a clip.

How many people can I reach with a broadcast?

That depends entirely on your Helix Universal Server and the network bandwidth it has available. For large broadcasts, you can use a network of Helix Universal Servers to reach thousands of RealOne Players.

Can RealNetworks broadcast clips for me?

Yes. Real Broadcast Network (RBN) offers a wide range of services for hosting broadcasts. Learn more about RBN at:

<http://www.realnetworks.com/products/rbn/index.html>

Getting Technical Support

RealNetworks offers a range of technical support features and documentation.

How do I get technical support from RealNetworks?

RealNetworks Technical Support operates an extensive Web site at **<http://service.real.com>**. The site includes answers to frequently asked questions, a documentation library, and a searchable knowledge base. To place a service call with Technical Support, fill out the e-mail form at the following Web page:

http://customerrelations.real.com/scripts/rnforms/contact_tech_service.asp

Where can I find additional documentation?

RealNetworks Technical Support maintains a documentation library at **<http://service.real.com/help/library/index.html>**. Most documents are available as bundled HTML archives that you can download, uncompress, and read with a Web browser. Many documents are also available in PDF format, which is suitable for printing. To read PDF files, you need Adobe's Acrobat Reader, which is available from Adobe's Web site:

<http://www.adobe.com/products/acrobat/readstep.html>

Where should I go for the latest information?

The RealNetworks Resources area is the main information site for content authors and software developers working with RealNetworks products. You can find it at the following Web address:

<http://www.realnworks.com/resources/index.html>

SMIL SYNTAX

This appendix explains the structure of SMIL file, as well as the syntax rules for writing SMIL markup. Understanding these rules will help you to avoid the common problems that media authors encounter when developing SMIL presentations.

SMIL Extension and File Names

A SMIL file is a simple text file that you can create with any text editor. As with a Ram file, make sure that you can save your output as plain text. Your SMIL file should have a simple name without spaces, and end with the extension `.smil`, as in `my_smilfile.smil`.

Tags, Attributes, and Values

SMIL tags that come in pairs follow this form:

```
<tag attribute="value"...> ... </tag>
```

There are three basic parts to these SMIL tags:

- | | |
|------------------|---|
| <i>tag</i> | In the first tag of a pair, the tag name comes just after a left angle bracket. Some tags may consist of just the name, as in the <code><body></code> tag. Other tags may have attributes. The last tag of a pair consists of just the tag name preceded by a slash, as in <code></smil></code> . A closing tag never has attributes. |
| <i>attribute</i> | Each attribute defines one aspect of the tag. If a tag has several attributes, the order of attributes doesn't matter. Attributes are typically predefined for a tag. The <code><smil></code> tag can include certain attributes, for example, while a <code><seq></code> tag can include different attributes. You'll get an error message from RealOne Player if you include an attribute that doesn't work with a tag. |

"value" Most SMIL attributes include an equals sign (=) followed by a value that must **always** be enclosed in double quotation marks. In some instances, you choose from a list of predefined values. In other cases, you define your own value. Values may be integers, percentages, names, and so on, depending on what types of values are appropriate for the attribute.

End Tags for Tag Pairs

If you're familiar with HTML, you know that browsers often let you be sloppy with end tags. For example, paragraphs in HTML should come between `<p>` and `</p>` tags, but most browsers allow you to leave out the closing `</p>` tag. SMIL doesn't let you get away with sloppiness, though, and you'll get an error message if you leave out a closing tag.

Closing Slash for Single Tags

If the tag is not part of a tag pair, as in `<head>` and `</head>`, it **must** close with a forward slash, as in `<meta/>`. The forward slash tells RealOne Player that everything it needs to know about this tag is contained within the tag's angle brackets, and it doesn't need to hunt for a closing tag. In general, tag pairs create sections, such as the header section or body section, while single tags describe specific actions, such as "display this title," or "play this video clip."

Tag and Attribute Case Sensitivity

Tag names, attributes, and many values are case-sensitive. This is another area where SMIL, unlike HTML, is strict. In HTML, for example, a paragraph tag can be `<p>` or `<P>`. But in SMIL, the `<smil>` tag cannot be `<SMIL>`, `<Smil>`, `<smIL>`, or any variation except entirely lowercase. Just make sure that you write tags and attributes exactly as they appear in this guide.

Tag ID Values

Any SMIL tag can have an ID in the form `id="value"`. Some SMIL tags require IDs. For example, each region in the layout requires an ID that you use to assign clips to play in the region. For other tags, IDs are optional depending on whether another SMIL element interacts with that tag. The following are rules and suggestions that apply to the IDs of all SMIL tags:

- All IDs for all tags in a SMIL file must be unique. If you define several `<region/>` tags, for example, each tag must have a unique ID. No `<region/>` tag can have the same ID as a `<video/>` tag, for instance.
- As with all SMIL values, IDs are case-sensitive. The attributes `id="videoregion"` and `id="videoRegion"` are different, for example. It is a good idea to follow a consistent practice, such as always making IDs lowercase.
- Do not use words separated by spaces in an ID. If you use two or more words for an ID, combine the words, or separate the words with an underscore or hyphen, as in `videoregion`, `video-region`, or `video_region`.
- The first character for an ID can be a letter, a colon, or an underscore. It cannot be a number or a special character such as an ampersand. You can use numbers and special characters after the first character, however. For example, you can use `id="video3"` as an ID, but not `id="3video"`.
- There is no minimum or maximum length for IDs.
- You may find it convenient to adopt a system for specifying IDs. You might use the suffix `_region` for all region IDs, for example.

The SMIL Tag

The first and last lines of a SMIL 2.0 file for RealOne Player are these:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
...
</smil>
```

SMIL 2.0 Namespace

The `<smil>` tag has one long, necessary attribute:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language" ...>
```

This attribute indicates that the file contains SMIL 2.0 markup. If you leave this attribute out and use just the tag name (`<smil>`), you're defining a SMIL 1.0 file, and many of the features described in this guide won't work. In other words, you must include this attribute and value **exactly** as shown above for a SMIL 2.0 file to work.

Note: Although the SMIL 2.0 and RealNetworks extension namespaces are valid HTTP URLs (you can open them in your browser), RealOne Player does not request these Web pages when it plays a SMIL 2.0 file.

RealNetworks Extension Namespace

To use SMIL 2.0 features customized for RealOne Player, you need to include the RealNetworks namespace in the <smil> tag:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"  
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
```

Customized attributes such as rn:sendTo must share the rn: prefix used by this namespace. Although you do not have to declare this namespace if you do not use the customized attributes, it is safe to include this namespace in any SMIL 2.0 file, even those played by SMIL-based players other than RealOne Player. If a SMIL-based player does not recognize a customized namespace and its associated attributes, it simply ignores them.

The Header Section

In a SMIL file, a header section typically follows the <smil> tag. The header section falls between <head> and </head> tags. Unlike the <smil> tag, the <head> tag never includes attributes:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"  
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">  
  <head>  
    <meta name="title" content="My First SMIL File"/>  
    <meta name="author" content="Pat Morales"/>  
    <meta name="copyright" content="(c)2001 Spectacular Media Limited"/>  
    <layout>  
      <root-layout width="320" height="240"/>  
      <region id="video_region"/>  
    </layout>  
  </head>  
  ...body section...  
</smil>
```

The Body Section

The body section is the companion to the header section, following it in the SMIL markup. Together, the body and header define all of your presentation features. Whereas the header describes the presentation's form, the body delivers the content. Like the header section, the body section has a start tag and an end tag, `<body>` and `</body>`, that do not take any attributes:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  ...header section...
  <body>
    <seq>
      <!-- The following clips play in sequence. -->
      <audio src="rtsp://helixserver.example.com/song1.rm"/>
      <audio src="rtsp://helixserver.example.com/song2.rm"/>
      <audio src="rtsp://helixserver.example.com/song3.rm"/>
    </seq>
  </body>
</smil>
```

Indentation and Line Returns

In the examples in this guide, each tag starts on a new line, and the entire file uses indentation, which is highly recommended for all SMIL markup. For example, the first lines of a SMIL file might look like this:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    <meta name="title" content="My First SMIL File"/>
```

Each line is indented from the preceding line to indicate that the new tag is part of the preceding section. The `<meta/>` tag is part of the `<head>` section, and the `<head>` section is part of the `<smil>` section.

Savvy authors always indent their markup to clarify the file's organization, typically pressing **Tab** once for each level of indentation. The only purpose for this is to help you, and anyone else reading your file, to understand how the file works. RealOne Player doesn't care about tabs and line returns, though. It has no problem figuring out how the following file works, but can you say the same?

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"><head><meta
name="title" content="My First SMIL File"/><meta name="author" content="Pat
Morales"/><meta name="copyright" content="(c)2001 Spectacular Media
Limited"/></head><body><seq><!-- The following clips play in sequence. -->
<audio src="song1.rm"/><audio src="song2.rm"/><audio src="song3.rm"/></seq>
</body></smil>
```

Comments

In a SMIL file, you can add a comment like the following:

```
<!-- The following clips play in sequence. -->
```

Note that this tag does not require a closing slash. If you know HTML, you've probably guessed why this is: this isn't a SMIL tag at all. This is an HTML-style comment tag. This tag is so widely used in HTML that SMIL adopted it wholesale.

As with indentation, comment tags don't affect your presentation. RealOne Player ignores them, and their only purpose is to annotate your file. Even with indentation, complex SMIL files can be hard to figure out. Comments let you describe what's going on in the file, and can be extremely helpful for explaining markup to others, or even reminding yourself of how the file works. You can even comment out whole sections of SMIL markup if you don't want RealOne Player to play that section.

Note, though, that the hyphens in a comment tag aren't decoration. The comment must start with these four characters:

```
<!--
```

and end with these three characters:

```
-->
```

Between these begin and end marks, the comment can be as long as you like. You cannot put one comment inside another comment, though.

Summary of SMIL Syntax

- To create SMIL 2.0 files, the `<smil>` tag must declare the SMIL 2.0 namespace:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
```


- Many attributes specific to RealOne Player use an `rn:` prefix, as in `rn:sendTo`. Using these attributes requires that you also declare the RealNetworks customizations namespace in the `<smil>` tag:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"  
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
```
- A SMIL header section, defined between `<head>` and `</head>` tags is optional, though highly recommended.
- The SMIL body section, defined between `<body>` and `</body>` tags, is required.
- Tags and attributes are case-sensitive, so you need to write them exactly as shown in this guide.
- Attribute values must be enclosed in double quotation marks.
- All tags either must be paired with an end tag, as in `<body>` and `</body>`, or must close with a forward slash, as in `<audio/>`.
- All id attribute values within a SMIL file must be unique. An id value cannot start with a number.
- Use indentation to clarify how your SMIL file is organized.
- Use HTML-style comments to annotate your SMIL file:

```
<!-- This is a comment. -->
```


SMIL TAG SUMMARY

This appendix summarizes the SMIL 2.0 tags and attributes described in this introductory guide. Be sure to familiarize yourself with “Conventions Used in this Guide” on page 4, which explains the typographical conventions used in this appendix. For a comprehensive list of SMIL 2.0 tags and attributes, see the SMIL tag summary in *RealNetworks Production Guide*.

<smil>...</smil>

The <smil> and </smil> tags must start and end the SMIL markup. The SMIL 2.0 namespace declaration is required. You must declare the RealNetworks extension namespace if your SMIL file includes a customized attribute that uses the rn: prefix.

<smil> Tag Namespaces

Namespace	Features Defined	Reference
xmlns="http://www.w3.org/2001/SMIL20/Language"	SMIL 2 Language Profile	page 117
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions"	RealNetworks extensions	page 118

Example

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
xmlns:rn="http://features.real.com/2001/SMIL20/Extensions">
  ...all additional SMIL 2.0 markup...
</smil>
```

Header Tags

The SMIL file header, created between <head> and </head> tags, contains tags that let you define the presentation’s layout and information. For instructions on defining the SMIL file header, see “SMIL File Basics” on page 60.

<meta/>

The header region's <meta/> tags provide presentation information. The content and name attributes are required for each <meta/> tag. For basic information about the <meta/> tag, see "Presentation Information" on page 61.

<meta/> Attributes

Attribute	Value	Function	Reference
content	<i>text</i>	Provides the content for the name attribute.	page 61
name	author	Lists the presentation author's name.	page 61
	copyright	Supplies the presentation copyright.	page 61
	title	Gives the presentation title.	page 61

Examples

```
<meta name="author" content="Jane Morales"/>
<meta name="title" content="Multimedia My Way"/>
<meta name="copyright" content="(c)2001 Jane Morales"/>
```

<layout>...</layout>

The <layout> and </layout> tags within the SMIL header contain other tags that define the layout of visual clips. Within the layout section, you define a root-layout area and separate regions for clips.

<root-layout/>

Within the layout section, a single <root-layout/> tag sets the overall size of the media playback pane. Clips play in regions created within the root-layout area. They do not play in the root-layout area directly. The height and width attributes are required for the <root-layout/> tag. For basic information about <root-layout/>, see "Setting the Media Pane Size" on page 79.

<root-layout/> Attributes

Attribute	Value	Default	Function	Reference
backgroundColor	<i>color_value</i>	black	Sets the window background color.	page 88
rn:contextWindow	auto openAtStart	auto	Opens the related info pane.	page 81
height	<i>pixels</i>	0	Sets the main window height.	page 80
width	<i>pixels</i>	0	Sets the main window width.	page 80

Example

```

<layout>
  <root-layout backgroundColor="maroon" width="320" height="240"/>
  <region ...playback region defined.../>
  <region ...playback region defined.../>
</layout>

```

<region/>

Following <root-layout/>, <region/> tags define the size, placement, and properties of each region used to play clips. A unique id attribute is required for each <region/> tag. For basic information about the <region/> tag, see “Creating Playback Regions” on page 81.

<region/> Attributes

Attribute	Value	Default	Function	Reference
backgroundColor	inherit transparent <i>color_value</i>	transparent	Sets the region background color.	page 88
bottom	auto <i>pixels</i> <i>percentage</i>	auto	Sets the region’s offset from the bottom of the pane.	page 83
fit	fill hidden meet scroll slice	hidden	Controls how clips fit the region.	page 92
height	auto <i>pixels</i> <i>percentage</i>	auto	Sets the region’s height.	page 83
id	<i>name</i>	(none)	Creates an ID for assigning clips.	page 82
left	auto <i>pixels</i> <i>percentage</i>	auto	Sets the region’s offset from the pane’s left side.	page 83
right	auto <i>pixels</i> <i>percentage</i>	auto	Sets the region’s offset from the pane’s right side.	page 83
showBackground	always whenActive	always	Determines when the background color appears.	page 89
top	auto <i>pixels</i> <i>percentage</i>	auto	Sets the region’s offset from the top of the pane.	page 83
width	auto <i>pixels</i> <i>percentage</i>	auto	Defines the region width.	page 83

Example

```

<layout>
  <root-layout .../>
  <region id="video_region" top="5" left="5" width="240" height="180"
    backgroundColor="blue" showBackground="whenActive"/>
</layout>

```

Clip Source Tags

You add clips to a presentation with one of the following source tags:

<audio/>	audio clip such as RealAudio
	image file in GIF, JPEG, or PNG format
<ref/>	any type of clip not covered by the other tags
<video/>	video clip such as RealVideo

The choice of tag does not affect playback. All clip source tags can use <ref/>, for example. The src attribute is required for all clip source tags. For basic information, see “Clip Source Tags” on page 62.

Clip Tag Attributes

Attribute	Value	Default	Function	Reference
author	<i>text</i>	(none)	Lists the clip’s author.	page 73
begin	<i>time_value</i>	0s	Delays normal playback time.	page 75
bitrate	<i>bits_per_second</i>	12288	Sets an image’s streaming speed in a <param/> tag.	page 97
clipBegin	<i>time_value</i>	0s	Specifies the clip’s internal timing mark where playback begins.	page 76
clipEnd	<i>time_value</i>	(none)	Specifies the clip’s internal timing mark where playback ends.	page 76
copyright	<i>text</i>	(none)	Lists the copyright for the clip.	page 73
dur	<i>time_value</i>	(none)	Sets the total time the clip plays.	page 76
fill	auto freeze remove	auto	Sets the fill state when the clip ends.	page 96
id	<i>name</i>	(none)	Provides the clip ID.	page 116
regAlign	topLeft topMid topRight midLeft center midRight bottomLeft bottomMid bottomRight	topLeft	Specifies which part of the clip aligns to the registration point.	page 90
region	<i>region_ID</i>	(none)	Assigns the clip to a region.	page 95
regPoint	topLeft topMid topRight midLeft center midRight bottomLeft bottomMid bottomRight	(none)	Specifies a point on the region.	page 90

(Table Page 1 of 2)

Clip Tag Attributes (continued)

Attribute	Value	Default	Function	Reference
src	<i>URL</i>	(none)	Provides a full or relative URL.	page 62
title	<i>text</i>	(none)	Provides a title for the clip.	page 73

(Table Page 2 of 2)

Examples

```
<video id="video1" src="rtsp://helixserver.example.com/media/video2.rm" region="video_region"
  begin="40s" clipBegin="5100ms" clipEnd="4.5min" fill="freeze"/>
```

```

  <param name="bitrate" value="5000"/>
</img>
```

<seq>...</seq>

The <seq> and </seq> tags play the enclosed clips in sequence. No attributes are required for a <seq> tag, which is described in “Playing Clips in Sequence” on page 71.

<seq> Attributes

Attribute	Value	Default	Function	Reference
begin	<i>time_value</i>	0s	Delays the normal group playback.	page 75
dur	<i>time_value</i>	(none)	Sets the total time the group plays.	page 76
id	<i>name</i>	(none)	Provides a group ID.	page 116

Example

```
<seq begin="30s">
  <audio src="rtsp://helixserver.example.com/one.rm"/>
  <audio src="rtsp://helixserver.example.com/two.rm"/>
</seq>
```

<par>...</par>

The <par> and </par> tags make enclosed clips play at the same time. No attributes are required for a <par> tag, which is described in “Playing Clips in Parallel” on page 95.

<par> Attributes

Attribute	Value	Default	Function	Reference
author	<i>text</i>	(none)	Lists an author for the group.	page 73
begin	<i>time_value</i>	0s	Delays the normal playback time.	page 75
copyright	<i>text</i>	(none)	Lists the copyright for the group.	page 73
dur	<i>time_value</i>	(none)	Sets the total time the group plays.	page 76
endsync	first ID last	last	Determines when the group ends.	page 95
id	<i>name</i>	(none)	Provides a group ID.	page 116
title	<i>text</i>	(none)	Lists a title for the group.	page 73

Examples

```
<par title="Song of the Day">
  <video src="rtsp://helixserver.example.com/newsong.rm"/>
  <ref src="rtsp://helixserver.example.com/newsong.rt"/>
</par>

<par endsync="vid1">
  <video id="vid1" src="rtsp://helixserver.example.com/video1.rm" region="video_region"/>
  <ref src="rtsp://helixserver.example.com/moreinfo.rt" region="text_region"/>
</par>
```

<area/>

An <area/> tag can define a link to an HTML page that opens automatically, or when the viewer clicks a clip. It fits within a clip source tag pair:

```
<video ...>
  <area .../>
</video>
```


The following table lists possible <area/> tag attributes. No attributes are required for this tag, but href is typically included. For basic information about the <area/> tag, see “Opening HTML Pages with SMIL” on page 64.

<area/> Attributes

Attribute	Value	Default	Function	Reference
actuate	onLoad onRequest	onRequest	Determines whether or not the link requires user activation.	page 65
alt	<i>text</i>	(none)	Supplies alternate text for the link.	page 65
begin	<i>time_value</i>	0s	Sets when the link becomes active.	page 75
dur	<i>time_value</i>	(none)	Sets the total time the link is active.	page 76
external	true false	false	Sends the link to the browser if true.	page 65
height	<i>pixels</i>	media height	Sets related info pane size in <param>.	page 66
href	<i>URL</i>	(none)	Gives the link URL.	page 65
id	<i>name</i>	(none)	Defines the tag ID.	page 116
rn:sendTo	_osdefaultbrowser _rpbrowser _rpcontextwin	(none)	Specifies a browser window that opens the HTML page.	page 66
sourcePlaystate	pause play stop	pause	Sets the clip state as the link opens.	page 66
width	<i>pixels</i>	330	Sets related info pane size in <param>.	page 66

Example

```
<video src="rtsp://helixserver.example.com/video.rm" region="video_region">
  <area href="http://www.example.com/context.html" external="true" rn:sendTo="_rpcontextwin"
    sourcePlaystate="play">
    <rn:param name="width" value="320"/>
    <rn:param name="height" value="240"/>
  </area>
</video>
```


RAM FILE SUMMARY

This appendix summarizes Ram file parameters, which are described in Chapter 3. Be sure to familiarize yourself with “Conventions Used in this Guide” on page 4, which explains the typographical conventions used in this appendix.

Parameter Syntax

A Ram file is plain text file that uses the file extension `.ram`. Each line holds the full URL to a clip or SMIL presentation, and can include parameters that affect playback. Separate the first parameter from the URL with a question mark (`?`), as shown here:

```
rtsp://helixserver.example.com/video1.rm?parameter=value
```

To set two or more parameters for the same clip, precede the second and all subsequent parameters with ampersands (`&`) instead of question marks:

```
rtsp://helixserver.example.com/video1.rm?parameter=value&parameter=value&parameter=value...
```

Parameters and Values

Ram File Parameters and Values

Parameter	Value	Default	Function	Reference
author	<i>text</i>	(none)	Indicates the clip author.	page 43
clipinfo	title= <i>text</i> artist name= <i>text</i> album name= <i>text</i> genre= <i>text</i> copyright= <i>text</i> year= <i>text</i> cdnum= <i>number</i> comments= <i>text</i>	(none)	Gives extended clip information.	page 44
copyright	<i>text</i>	(none)	Gives the copyright notice	page 43
end	<i>hh:mm:ss.x</i>	(none)	Ends the clip at the specified point.	page 41

(Table Page 1 of 2)

Ram File Parameters and Values (continued)

Parameter	Value	Default	Function	Reference
mode	normal theater toolbar	normal	Opens RealOne Player in one of three initial playback modes.	page 41
rpcontextheight	<i>pixels</i>	media height	Sets the related info pane's height.	page 38
rpcontextparams	<i>parameters</i>	(none)	Adds parameters to rpcontexturl.	page 38
rpcontexttime	<i>dd:hh:mm:ss.x</i>	0	Specifies a time at which an HTML page displays in the related info pane.	page 38
rpcontexturl	<i>URL _keep</i>	(none)	Displays the specified URL in the related info pane.	page 38
rpcontextwidth	<i>pixels</i>	330	Sets the related info pane width.	page 38
rpurl	<i>URL</i>	(none)	Gives a URL for the media browser.	page 38
rpurlparams	<i>parameters</i>	(none)	Appends parameters to rpurl.	page 38
rpurltarget	<i>_rpbrowser name</i>	<i>_rpbrowser</i>	Sets the target for rpurl as the media browser pane or a secondary window.	page 39
rpvideofillcolor	<i>color_value</i>	black	Sets the media playback pane color.	page 39
screensize	double full original	original	Sets the size at which the clip or presentation opens.	page 41
showvideocontrolsoverlay	0 1	1	Hides the video controls overlay in the media playback pane when 0.	page 41
start	<i>hh:mm:ss.x</i>	0	Starts the clip at the specified point.	page 41
title	<i>text</i>	(none)	Specifies the clip title.	page 43

(Table Page 2 of 2)

Examples

```
rtsp://helixserver.example.com/video1.rm?rpcontextheight=250
```

```
&rpcontextwidth=280&rpcontexturl="http://www.example.com/relatedinfo1.html"
```

```
rtsp://helixserver.example.com/video2.rm?rpurl="http://www.example.com/index.html"
```

```
rtsp://helixserver.example.com/sample1.smil?screensize=full
```

```
rtsp://helixserver.example.com/audio1.rm?start=55&end=1:25
```

```
rtsp://helixserver.example.com/introvid.rm?title="Introduction to Streaming Media  
Production"&author="RealNetworks, Inc."&copyright="&#169;2001, RealNetworks, Inc."
```

```
rtsp://helixserver.example.com/song1.rm?clipinfo="title=Artist of the Year|artist name=Your Name  
Here|album name=My Debut|genre=Rock|copyright=2001|year=2001|comments=This one really  
knows how to rock!"
```

GLOSSARY

B **bandwidth**

The upper limit on the amount of data, typically expressed as kilobits per second (Kbps), that can pass through a network connection.

binary tag

A SMIL tag that comprises opening and closing tags, such as <ref> and </ref>. Many unary tags can become binary tags when necessary to enclose other tags.

bit

The smallest unit of measure of data in a computer. A bit has a binary value, either 0 or 1.

bit rate

A measure of bandwidth, expressed as the number of bits transmitted per second. A 28.8 Kbps modem, for example, can transmit or receive around 29,000 bits per second.

blank time

A period during a presentation in which RealOne Player is not paused, but no activity occurs onscreen. You typically insert blank time with the SMIL begin attribute.

broadcast

To deliver a presentation, whether live or prerecorded, in which all viewers join the presentation in progress. Contrast to *on-demand*.

buffering

The receiving and storing of data before it is played back. A clip's initial buffering is called *preroll*. After this preroll, excessive buffering may stall the presentation.

byte

A common measurement of data. One byte consists of 8 bits.

C cable modems

Devices that allow rapid transmission and reception of data over television cable. They are digital devices, unlike dial-up modems, which transmit analog data.

camel case

A capitalization convention in which words in a phrase are joined, and each word after the first begins with a capital letter. SMIL 2.0 attributes and values generally use camel case, as in `whenNotActive`.

client

A software application that receives data from a server. A Web browser is a client of a Web server. RealOne Player is a client of Helix Universal Server.

clip

A media file within a presentation. Clips typically have an internal timeline, as with RealAudio and RealVideo.

codec

Coder/decoder. Codecs convert data between uncompressed and compressed formats, reducing the bandwidth a clip consumes.

D download

To send a file over a network with a nonstreaming protocol such as HTTP. Contrast to *stream*.

DSL

Digital Subscriber Line. A technology for transmitting digital data over a regular telephone line much faster than through dial-up modems.

duress stream

A low-bandwidth SureStream audio or video stream that Helix Universal Server uses if a connection's available bandwidth drops greatly.

E encoding

Converting a file into a compressed, streaming format. For example, you can encode WAV files as RealAudio clips.

F Flash

A software application and an animation format created by Macromedia. RealOne Player can play Flash animations and stream them in parallel with other clips, such as RealAudio clips.

Flash Player file

A compressed Flash file format (file extension .swf) suitable for streaming. To stream Flash, you export the Flash Player file and tune it so that it plays well in RealOne Player.

fps

Frames Per Second. The number of video frames that displays each second in a streaming video clip.

H**Helix**

A general term for the technology on which RealNetworks products are based. The RealNetworks suite for Helix includes Helix Universal Server and Helix Producer.

Helix Producer

The primary tool for encoding RealAudio and RealVideo clips.

Helix Universal Server

Server software used to stream multimedia presentations to RealOne Player and other media players.

HTTP

Hypertext Transport Protocol. The protocol used by Web servers to communicate with Web browsers. In contrast, Helix Universal Server streams clips to RealOne Player with RTSP.

I**ISDN**

Integrated Services Digital Network. Technology that makes digital data connections at 64 or 112 Kbps possible over telephone lines.

ISP

Internet Service Provider. A company that provides access to the Internet. Some ISPs have Helix Universal Server available to stream media clips.

K**kilobit (Kb)**

A common unit of data measurement equal to 1024 bits. A kilobit is usually referred to in the context of bit rate per unit of time, such as kilobits per second (Kbps).

kilobyte (KB)

A common unit of data measurement equal to 1024 bytes or 8 kilobits.

- L LAN**
Local Area Network. A computer network confined to a local area, such as a single building. LANs vary in speed, with bandwidth shared among all networked devices.
- M metafile**
Another name for a Ram file.
- MPEG**
A standards-based audio and video format that you can stream to RealOne Player. You can use MPEG-1 and MPEG-4 video, or the MP3 audio format.
- N namespace**
A declaration that identifies the features used in a SMIL presentation. For SMIL 2.0 and higher, the <smil> tag must declare a namespace.
- O on-demand**
A type of streaming in which a clip plays from start to finish when a user clicks a link. Most clips are streamed this way. Contrast to *broadcast*.
- P preroll**
Buffering that occurs just before a clip plays back. Preroll should be no more than 15 seconds.
- presentation**
A clip or group of clips streamed from Helix Universal Server to RealOne Player. The presentation can also include HTML URLs that open in the RealOne Player HTML panes.
- R Ram file**
A text file that uses the file extension `.ram`. It launches RealOne Player and gives it the URL to a streaming clip or presentation.
- RealAudio**
A clip type for streaming audio over a network. RealAudio clips use the `.rm` extension.
- RealOne Player**
The successor to RealPlayer 8, RealOne Player combines streaming and digital download technologies. It supports the SMIL 2.0 and 1.0 standards.

RealPix

A clip type (file extension .rp) for streaming still images over a network. RealPix uses a markup language for creating special effects such as fades and zooms.

RealPlayer G2

The RealNetworks client software that introduced plug-ins and the ability to update itself. It, along with the later RealPlayer 7 and RealPlayer 8, supports the SMIL 1.0 standard.

RealSlideshow

A RealNetworks tool for creating streaming slideshows based on the RealPix markup.

RealText

A clip type (file extension .rt) for streaming text over a network. It uses a markup language for formatting text.

RealVideo

A clip type for streaming video over a network. RealVideo clips use the extension .rm.

rebuffering

An undesirable state in which RealOne Player must pause a presentation to wait for streaming data to arrive. Rebuffering can result from network conditions, or a poorly produced presentation.

RTSP

Real-Time Streaming Protocol. An open, standards-based control protocol that Helix Universal Server uses to stream clips to RealOne Player or any RTP-based client. Contrast to *HTTP*.

S server

1. A software application, such as a Web server or Helix Universal Server, that sends requested data over a network.
2. A computer that runs server software.

SMIL

Synchronized Multimedia Integration Language. A markup language for specifying how and when each clip plays within a presentation. SMIL files use the extension .smil.

stream

1. To send a media clip over a network so that it begins playing back as quickly as possible.

2. A flow of a single type of data, measured in kilobits per second (Kbps). A RealVideo clip's soundtrack is one stream, for example.

SureStream

A RealNetworks technology that enables a RealAudio or RealVideo clip to stream at multiple bit rates.

U unary tag

A SMIL tag that includes a closing slash, as in <ref/>. Many unary tags can become binary tags when necessary to enclose other tags.

URL

Uniform Resource Locator. A location description that enables a Web browser or RealOne Player to receive a clip stored on a Web server or Helix Universal Server.

INDEX

- A**
 - accessibility attributes in SMIL, 109
 - ActiveX, 17
 - actuate attribute, 65
 - advertising on RealGuide, 105
 - alt attribute in links, 65
 - animation
 - Flash, 23
 - SMIL, 109
 - <area/> tag, 64
 - see also* linking
 - audio
 - descriptions for the sight-impaired, 109
 - getting high quality, 104
 - visualizations in RealOne Player, 11
 - <audio/> tag, 62
 - author attribute
 - in events file, 51
 - in Javascript, 56
 - in Ram file, 43
 - in SMIL, 73
- B**
 - background color
 - in Javascript, 57
 - in Ram file, 39
 - in SMIL, 88
 - backwards compatibility, 18
 - bandwidth
 - leaving for other processes, 29
 - multiclip presentations, 29
 - network connection speeds, 25
 - overview, 24
 - SMIL-based choices, 109
 - SureStream clips, 25
 - begin attribute, 75
 - bit rate, *see* bandwidth
 - bitrate parameter for images, 97
 - bottom attribute in <region/> tag, 83
 - broadcasting
 - live vs. prerecorded, 112
 - RealGuide listings, 105
 - required equipment, 111
 - SMIL, 112
 - stream limit, 112
 - SureStream, 111
 - through ISP, 111
 - through RealNetworks, 112
 - Web servers, 112
 - brush objects, 77
- C**
 - cable modem bandwidth targets, 25
 - caching content, 13
 - captions, 109
 - centering clips in regions
 - example, 100
 - registration point for, 90
 - clip caching, 13
 - clip source tags
 - absolute local paths, 64
 - base URL, 77
 - HTTP URLs, 35
 - relative local paths, 63
 - RTSP URLs, 35
 - writing, 63
 - clip transfer to servers, 47
 - clipBegin attribute, 76
 - clipEnd attribute, 76
 - clipinfo parameter
 - in events file, 51
 - in Javascript, 56
 - in Ram file, 44
 - colors, *see* background color
 - comparison of production techniques, 17

- context pane, *see* related info pane
- contextWindow attribute, 81
- copyright attribute
 - in events file, 51
 - in Javascript, 56
 - in Ram file, 43
 - in SMIL, 73
- copyright protection, 104
- D**
 - digital rights management, 104
 - documentation library, 4
 - double-size mode
 - overview, 12
 - setting, 40
 - download icon for RealOne Player, 105
 - downloading data before clips play, 77
 - DSL bandwidth targets, 25
 - dur attribute, 76
 - with endsync attribute, 98
- E**
 - embedding media in HTML pages, 7
 - encoding tools, *see* Helix Producer
 - end attribute in SMIL, 77
 - end parameter in Ram file, 41
 - endsync attribute, 95
 - with dur attribute, 98
 - escape codes, 45
 - events file
 - creating, 50
 - merging, 52
 - example files
 - playback problems, 3
 - where to find, 3
 - external attribute, 65
- F**
 - file transfer to servers, 47
 - file:// in URLs, 36
 - fill attribute, 96
 - fit attribute
 - affect on clips, 93
 - clip scaling recommendations, 94
 - filling the region, 94
 - illustration of effects, 94
 - maintaining aspect ratio, 94
 - maintaining clip size, 94
 - Flash animation, 23
 - freezing a clip onscreen, 96
 - FTP, 47
 - full-screen mode
 - overview, 12
 - setting, 40
- G**
 - GIF images, 23
 - graphics
 - formats, 23
 - modifying transparency, 77
 - streaming speed, 97
 - streaming time calculation, 98
 - groups
 - see* parallel groups
 - see* sequences
- H**
 - height attribute
 - <region/> tag, 83
 - <root-layout/> tag, 79
 - Helix Producer
 - download URL, 23
 - encoding from a camera, 104
 - input formats, 21, 106
 - overview, 22
 - SureStream clips, 25
 - video sizes, 27
 - Helix Server
 - administrator, 29
 - advanced features, 30
 - bandwidth constraints on, 30
 - broadcasting, 111
 - download URL, 110
 - operating systems, 110
 - stream maximum, 30
 - through ISPs, 30
 - HTML Help version of this guide, 3
 - HTML pages
 - embedding media in, 7, 104
 - opening
 - with a Ram file, 37
 - with an events file, 50
 - with Javascript, 56
 - with SMIL, 64

- on a mouse click, 68
 - while a clip plays, 67
 - supported technologies, 103
 - see also* media browser pane
 - see also* related info pane
- HTML+Javascript version of this guide, 3
- HTTP
 - compared to RTSP, 36
 - in clip URLs, 35
- http:// in URLs, 35
- hyperlinking
 - see* HTML pages
 - see* linking
- I**
 - id attribute
 - case-sensitivity, 117
 - first character, 117
 - length, 117
 - spaces in, 117
 - uniqueness, 117
 - image maps, 69
 - images
 - formats, 23
 - modifying transparency, 77
 - streaming speed, 97
 - streaming time calculation, 98
 - tag, 62
 - Internet broadcasting, *see* broadcasting
 - ISDN bandwidth targets, 25
 - ISPs and Helix Server, 30
- J**
 - Javascript commands
 - background color, 57
 - clip information, 56
 - media browser pane link, 57
 - overview, 17
 - playing a clip, 56
 - related info pane link, 56
 - JPEG images, 23
- K**
 - keystroke activation of links, 69
- L**
 - LAN bandwidth use
 - lowering, 29
 - maximum, 25
 - language choices in SMIL, 109
 - launching RealOne Player, 33
 - laying out presentations, *see* regions
 - layout examples, 100
 - left attribute in <region/> tag, 83
 - letterbox videos, 100
 - linking
 - advanced features, 69
 - alternate text, 65
 - automatically opening a page, 65
 - clip volume adjustment, 69
 - clips to Ram file, 34
 - clips to SMIL file, 62
 - external attribute, 65
 - HTML pane selection, 66
 - image maps, 69
 - keystroke activation, 69
 - media clip to media clip, 69
 - media playback state, 66
 - Ram file to Web page, 47
 - SMIL examples
 - opening a page on a click, 68
 - opening several pages, 67
 - SMIL file to Ram file, 60
 - tabbing order of links, 69
 - logos for RealOne Player, 105
- M**
 - Macromedia Flash, 23
 - manuals, where to find, 4
 - max attribute, 77
 - media browser pane
 - Now Playing list, 14
 - opening
 - through events file, 50
 - through HTML page hyperlink, 16
 - through Javascript, 57
 - through Ram file, 37
 - through SMIL, 66
 - overview, 14
 - secondary windows, 14
 - supported technologies, 103
 - target name in hyperlinks, 16
 - URL parameters, 38
 - media playback pane

- background color
 - through Javascript, 57
 - through Ram file, 39
 - through SMIL, 88
 - hyperlinks to other media, 69
 - overview, 8
 - pop-up media windows, 102
 - sizing, 9
 - supported technologies, 8
 - with related info pane, 10
 - see also* SMIL
 - min attribute, 77
 - mode parameter in Ram file, 41
 - modem bandwidth targets, 25
 - MPEG, 21
- N** network connection speeds, 25
- Now Playing list, 14
- P** <par> tag, 95
- parallel groups
 - author information, 98
 - bandwidth issues, 97
 - defining, 95
 - delays through slow image streaming, 98
 - group endpoint
 - first clip, 95
 - last clip, 98
 - specific clip, 95
 - with sequences, 98
 - password authentication, 30
 - pay-per-view, 30
 - PDF version of this guide, 3
 - PNG images, 23
 - prefetching data, 77
 - publishing tools, *see* Helix Producer
- Q** QuickTime and SMIL, 21
- R** Ram file
 - browser pane URL, 37
 - clip end time, 40
 - clip playback size, 40
 - clip start time, 40
 - comments, 36
 - file extension, 35
 - line breaks, 35
 - linking
 - to Helix Server, 34
 - to local files, 34
 - to Web server, 34
 - overview, 15, 33
 - parameter syntax, 37
 - placement on servers, 47
 - RealOne Player start mode, 40
 - reasons for using, 33
 - related info pane URL, 37
 - sequences of clips, 35
 - syntax, 34
 - URLs, 35
 - with SMIL file, 60- RealGuide, 105
- RealOne Player
 - download logo, 105
 - subscription service, 103
- RealPix
 - description, 24
 - through RealSlideshow, 23
- RealSlideshow, 23
- RealText, 24
- RealVideo
 - quality guide, 27
 - recommended sizes, 27
- <ref/> tag, 62
- region attribute in clip source tags, 95
- regions
 - assigning to clips, 95
 - audio clips, 88
 - background colors, 88
 - inheriting, 88
 - transparency until clip plays, 89
 - bottom attribute, 83
 - clip scaling, 92
 - cropped at pane boundaries, 88
 - defining, 81
 - examples
 - centering a video, 100
 - four offsets defined, 84
 - letterbox clip, 100

- one offset defined, 86
 - overlapping regions, 87
 - side-by-side clips, 101
 - size and two offsets defined, 85
 - two offsets defined, 86
 - width and height defined, 84
- fit attribute, 92
- height attribute, 83
- id attribute, 82
- left attribute, 83
- percentage values, 83
- positions, 83
- reusing, 81
- right attribute, 83
- root-layout
 - defining, 79
 - sizing
 - double-screen mode, 80
 - example, 79
 - full-screen mode, 80
 - RealOne Player controls, 79
- sizes
 - pixels and percentages, 83
 - decimal percentages, 88
 - mixing, 88
 - setting, 83
 - subregions, 102
 - top attribute, 83
 - transparency, 88
 - volume adjustments, 102
 - width attribute, 83
 - z-index attribute, 102
- registration points
 - clip source tags, 90
 - common values, 91
 - defining in SMIL header, 102
 - methods of creating, 90
 - misalignment problems, 91
- regPoint attribute, 91
- related info pane
 - caching, 13
 - opening
 - through events file, 50
 - through Javascript, 56
 - through Ram file, 37
 - through SMIL, 66
 - overview, 12
 - sizing
 - browser pane width override, 13
 - media pane height override, 13
 - overview, 12
 - persistence, 13
 - through events file, 50
 - through Javascript, 57
 - through Ram file, 38
 - through SMIL, 66
 - supported technologies, 103
 - URL parameters, 38
 - with media playback pane, 10
 - removing a clip after it plays, 96
 - repeatCount attribute, 77
 - repeatdur attribute, 77
 - resize behavior of clips, 102
 - right attribute in <region/> tag, 83
 - RMEvents, 52
 - rn:sendTo attribute, 66
 - <root-layout/> tag, 79
 - see also* regions
 - rpcontextheight parameter, 38
 - rpcontextparams parameter, 38
 - rpcontexturl parameter, 38
 - rpcontextwidth parameter, 38
 - rpurl parameter, 38
 - rpurlparams parameter, 38
 - rpurltarget parameter, 39
 - rpvideofillcolor parameter, 39
 - RTSP
 - compared to HTTP, 36
 - in clip URLs, 36
 - in URLs, 35
 - overview, 36
 - rtsp:// in URLs, 36
- S**
 - sample files
 - playback problems, 3
 - where to find, 3
 - scaling clips in regions, 92
 - screenize parameter, 41
 - scroll bars in SMIL regions, 92
 - secondary media windows, 102

- sendTo attribute, 66
- <seq> tag, 71
- sequences
 - defining
 - through Ram file, 35
 - through SMIL, 71
 - Next Clip command, 72
 - seeking through, 72
 - single presentation vs. multiple clips, 72
 - with parallel groups, 98
- showvideocontroloverlay parameter in Ram file, 41
- slideshows, 23, 106
- SMIL 1.0, 108
- SMIL 2.0
 - animations, 109
 - case-sensitivity, 121
 - comments, 121
 - id attributes, 116
 - indentation, 121
 - namespaces
 - customizations, 118
 - standard, 117
 - prefetching, 77
 - quotation marks for values, 121
 - text display, 77
 - transition effects, 77
 - with Windows Media, 21
- sourcePlaystate attribute, 66
- special effects in SMIL, 77
- src attribute, 62
- start parameter in Ram file, 41
- streaming data before clips play, 77
- streaming speeds, 25
- subregions, 102
- subscription services, 103
- SureStream
 - bandwidth adjustment, 26
 - broadcasting, 111
 - overview, 25
 - target audiences, 27
 - with a Web server, 107
- switching in SMIL, 109

T

- tabbing order of links, 69
- target name for media browser pane, 16
- technical support, 5
- tenths of seconds display, 75
- text clips, 77
- theater mode, 40
- three-pane environment, 7
- timing in SMIL
 - advanced timing attributes, 78
 - begin attribute, 75
 - clipBegin attribute, 76
 - clipEnd attribute, 76
 - delaying clip playback, 75
 - dur attribute, 76
 - durations, 76
 - end attribute, 77
 - endsync attribute, 95
 - max attribute, 77
 - min attribute, 77
 - repeatCount attribute, 77
 - repeatDur attribute, 77
 - repeating clips, 77
- title attribute
 - in events file, 51
 - in Javascript, 56
 - in Ram file, 43
 - in SMIL, 73
- toolbar mode, 40
- top attribute in <region/> tag, 83
- transition effects in SMIL, 77
- transparency
 - clips, 89
 - modifying in clips, 77
 - regions, 88

U

- URL events
 - file for, 50
 - overview, 15
- URLs
 - to Helix Server, 34
 - to local files, 34
 - to Web server, 34
 - see also* linking

- V**
 - VBScript, 17
 - video quality, 104
 - <video/> tag, 62
 - visualizations, 11
 - volume level
 - for clips in regions, 102
 - when links open, 69

- W**
 - Web pages, *see* HTML pages
 - Web server
 - broadcasting, 112
 - clip delivery, 30
 - limitations, 30
 - SureStream, 107
 - Webcasting, *see* broadcasting
 - wide screen video display, 100
 - width attribute
 - <region/> tag, 83
 - <root-layout/> tag, 79
 - Windows Media and SMIL, 21

- Z**
 - z-index attribute, 102

