EMBEDDED REALPLAYER®
EXTENDED FUNCTIONALITY GUIDE

RealSystem™ G2

Revision Date: December 21, 1998

RealNetworks, Inc.
1111 Third Avenue, Suite 2900
Seattle, WA 98101 USA

http://www.real.com

CONTENTS

INTRODUCTION *1*

RealSystem Components 1
Conventions Used in this Manual 2
Technical Support 2
RealForum 3

1 USING JAVASCRIPT AND THE NETSCAPE PLUG-IN *4*

Naming a Plug-in Instance 4
Using RealPlayer Methods through JavaScript 4
Handling URLs through <EMBED> 5
Using PREFETCH with <EMBED> 5
Receiving Callbacks 6

2 USING VBSCRIPT AND THE ACTIVEX CONTROL *8*

Using RealPlayer Methods through VBScript 8
Handling URLs through <OBJECT> 9
Using PREFETCH with <OBJECT> 9
Receiving Callbacks 10

3 METHODS *12*

Playback Commands 12
PlayState Information 13
Attributes 13
Clip Position 17
Clip/TAC Information 18
Playlist/Multiclip 18
Live Playback 20
User Interface Components 20
Error Handling 21
Display State 23
Volume/Mute Control 23
Context Menu 24
Network Information 25
Version Information 26
Event Flags 26

User Preference Settings....26
RealPlayer 5.0 Compatibility....27

4 CALLBACK METHODS *28*

RealPlayer G2 Callbacks....28
RealPlayer 5.0 Compatibility....31

INTRODUCTION

RealPlayer G2 includes a Netscape plug-in and ActiveX control that expose RealPlayer functionality. This lets you play back a RealSystem presentation directly in a Web page without launching RealPlayer as a separate application. You simply add `<EMBED>` tag or `<OBJECT>` tag mark-up to a Web page. The embedded presentation typically includes an image window and one or more standard RealPlayer interface elements, such as the **Stop** button, the volume slider, and so on.

Additional Information

RealSystem G2 Production Guide available at **http://service.real.com/help/library/index.html** explains the basics of embedding a presentation.

Using a scripting language such as JavaScript or VBScript, you can use RealPlayer functions such as the stop, play, and volume controls without using RealPlayer's standard interface components. You can use your own graphic image for a Stop button, for example. When a user clicks the button image, a stop command is sent to RealPlayer's Netscape plug-in or ActiveX control. This document describes all the supported methods and events for extending RealPlayer's embedded functionality.

Tip

The HTML version of this guide, available at **http://service.real.com/help/library/index.html**, contains samples of embedded playback.

RealSystem Components

You need the following tools to create and test your embedded presentation:

- RealPlayer G2

  Use RealPlayer G2, available free at **http://www.real.com**, to test your JavaScript or VBScript extensions. Installing RealPlayer G2 also installs its

Netscape plug-in and ActiveX control for embedded playback. The RealPlayer G2 installation includes a ZIP file with Java classes for use with the Netscape plug-in.

- RealServer

  RealServer streams clips to RealPlayer. RealServer is not necessary for testing local playback of embedded clips, but is necessary for streaming presentations to RealPlayer G2 over a network. *RealServer Administration Guide* is available at **http://service.real.com/help/library/index.html**.

- RealSystem G2 Software Development Kit (SDK)

  The RealSystem G2 SDK is not necessary for extending RealPlayer's embedded playback features, but is required for advanced programming tasks such as building a new client interface on top of the RealPlayer G2 core. A knowledge of C++ programming is required to use the SDK. Register for and download the SDK at **http://www.real.com/devzone/**.

Conventions Used in this Manual

The following table explains the conventions used in this manual.

Notational Conventions

Convention	Meaning
variables	Italicized text represents variables. Substitute values appropriate for your situation.
[options]	Square brackets indicate optional values you may or may not need to use.
choice 1|choice 2	Vertical lines separate values you can choose between.
...	Ellipses indicate nonessential information omitted from the example.

Technical Support

For technical support with RealSystem G2, please fill out the form at:

- **http://service.real.com/contact/email.htm**

The information you provide in this form will help technical support personnel to give you a prompt response. For general information about RealNetworks' technical support, visit:

- **http://service.real.com/help/call.html**

RealForum

RealNetworks also encourages you to join RealForum, an e-mail discussion group about RealSystem products in which developers and content producers post tips and ask for assistance. RealNetworks employees monitor the postings and offer suggestions as appropriate. You can sign up for RealForum at **http://proforma.real.com/mario/devzone/realforum.html**.

USING JAVASCRIPT AND THE NETSCAPE PLUG-IN

You can use JavaScript with the RealPlayer G2 Netscape plug-in with the following browsers:

- Netscape Navigator 3.0, 4.0, and higher
- Microsoft Internet Explorer 3.0 and 4.0

Additional Information

RealSystem G2 Production Guide available at **http://service.real.com/help/library/index.html** explains the basics of embedding a presentation with the Netscape plug-in.

Naming a Plug-in Instance

To refer to an embedded instance with JavaScript, you include a `NAME` parameter in the `<EMBED>` tag:

```
<EMBED NAME=vid SRC="..." WIDTH=300 HEIGHT=134>
```

You can then refer to the instance through a JavaScript command such as this:

```
<Input Type="button" Value="play" onClick="document.vid.DoPlay()">
```

Using RealPlayer Methods through JavaScript

To extend RealPlayer's Netscape plug-in functionality with JavaScript, you first embed the source file in an HTML page with the `<EMBED>` tag:

```
<EMBED NAME=javademo
   SRC="demo.rpm"
   WIDTH=220 HEIGHT=180
   CONSOLE=one
```

```
CONTROLS=ImageWindow
BACKGROUNDCOLOR=white
CENTER=true
>
```

In the <EMBED> tag, the NAME parameter provides the name used by the JavaScript functions. For JavaScript to function with RealPlayer, the <EMBED> tag must **not** contain the parameter NOJAVA=true. That parameter prevents the browser's Java Virtual Machine from starting up.

You can then use JavaScript to issue RealPlayer commands to control the embedded presentation. The following example shows a simple form that provides a Play, Pause, and Stop button for the embedded presentation.

```
<FORM>
<INPUT TYPE="button" VALUE="Play" onClick="document.javademo.DoPlay()">
<INPUT TYPE="button" VALUE="Pause" onClick="document.javademo.DoPause()">
<INPUT TYPE="button" VALUE="Stop" onClick="document.javademo.DoStop()">
</FORM>
```

Additional Information

The section "Methods" beginning on page 12 lists the RealPlayer methods.

Handling URLs through <EMBED>

You can use the AUTOGOTOURL parameter in the <EMBED> tag to determine how URLs in the presentation are handled. The default value of true applies if you leave the parameter out. In this case any URL embedded in the presentation goes to the browser. If you set this parameter to false, RealPlayer sends the URL to the Java applet or application with the OnGotoURL() call.

Note

The SetAutoGoToURL() method (see page 15) can override the AUTOGOTOURL parameter. AUTOGOTOURL is not compatible with the RealPlayer 5.0 Netscape plug-in, however, so use SetAutoGoToURL() when planning backwards-compatibility.

Using PREFETCH with <EMBED>

The PREFETCH parameter causes RealPlayer G2 to get the stream description information before the stream starts. You can use this to find out the size and

width of an embedded clip, for example, then dynamically create the <EMBED> tag for the image window, using the clip's native size for the WIDTH and HEIGHT parameters. To do this, you would embed a control, such as the default control panel, and set PREFETCH to true:

```
<EMBED NAME=console
   SRC="demo.rpm"
   WIDTH=400 HEIGHT=100
   CONSOLE=one
   CONTROLS=All
   PREFETCH=true
   >
```

The control fetches the stream information, then responds with the callback OnPreFetchComplete() (see page 31). You next use GetClipWidth() and GetClipHeight() (see page 18) to determine the native size of the image window. Through a means such as DHTML, you can then create the image window control. If the width and height were 320 by 240, respectively, you could generate an <EMBED> tag like the following:

```
<EMBED NAME=image
   SRC="demo.rpm"
   WIDTH=320 HEIGHT=240
   CONSOLE=one
   CONTROLS=ImageWindow
   >
```

Receiving Callbacks

When using JavaScript and the Netscape plug-in, you use LiveConnect to receive callbacks. LiveConnect is described at **http://home.netscape.com/navigator/v3.0/liveconnect.html**. The java directory of the HTML version of this manual contains callback.java, which is a sample Java class used to create a Java callback object. You can modify this class as necessary to get notifications from RealPlayer's RMObserver or RAObserver interface, both of which are included in the rpcl3260.zip file installed in the RealPlayer folder:

- RMObserver.class

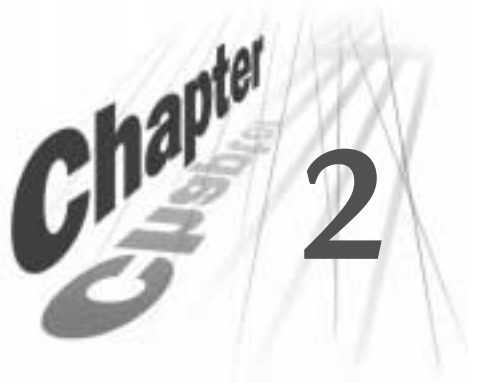
  RMObserver is a Java interface for events coming from RealPlayer G2 (but not RealPlayer 5.0). Any object implementing this interface may register itself into RealPlayer G2 to get a large set of notifications. To see how this is done, view the **adviseG2()** function in RAPlayer.java, which is included in the ZIP file.

- `RAObserver.class`

  RAObserver is a Java interface for events coming from RealPlayer 5.0. Any object implementing this interface may register itself into RealPlayer G2 or 5.0 to get a small set of notifications. To see how this is done, view the **advise()** function in `RAPlayer.java`, which is included in the ZIP file.

  Additional Information

  The section "CallBack Methods" beginning on page 28 lists the RealPlayer callback events.



USING VBSCRIPT AND THE ACTIVEX CONTROL

You can use VBScript with the RealPlayer G2 ActiveX control to provide playback capabilities within these products:

- Internet Explorer 3.0 and 4.0
- Any application that supports ActiveX controls, such as Visual Basic, Visual C++, Microsoft Access, and so on.

Additional Information

RealSystem G2 Production Guide available at **http://service.real.com/help/library/index.html** explains the basics of embedding a presentation with the ActiveX Control.

Using RealPlayer Methods through VBScript

To extend RealPlayer's ActiveX functionality on Internet Explorer, you first embed the source file in an HTML page with the `<OBJECT>` tag:

```
<OBJECT ID=RVOCX CLASSID="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA"
WIDTH=220 HEIGHT=180>
<PARAM NAME="SRC" VALUE="rtsp://realserver.company.com/media/animate.swf">
<PARAM NAME="CONSOLE" VALUE="one">
<PARAM NAME="CONTROLS" VALUE="ImageWindow">
<PARAM NAME="BACKGROUNDCOLOR" VALUE="white">
<PARAM NAME="CENTER" VALUE="true">
</OBJECT>
```

In the `<OBJECT>` tag, the `ID` parameter identifies the embedded clip for reference by VBScript parameters.

You can then use VBScript to issue RealPlayer commands to control the embedded presentation. The following example shows a simple form that provides a Play, Pause, and Stop button for the embedded presentation.

```
<FORM>
<input TYPE="button" VALUE="Play" NAME="doplay">
  <script LANGUAGE="VBScript" FOR="doplay" EVENT="onClick">
  RVOCX.DoPlay
  </script>
<input TYPE="button" VALUE="Pause" NAME="pause">
  <script LANGUAGE="VBScript" FOR="pause" EVENT="onClick">
  RVOCX.DoPause
  </script>
<input TYPE="button" VALUE="Stop" NAME="stop">
  <script LANGUAGE="VBScript" FOR="stop" EVENT="onClick">
  RVOCX.DoStop
  </script>
</FORM>
```

Additional Information

The section "Methods" beginning on page 12 lists the RealPlayer methods.

Handling URLs through <OBJECT>

You can use the AUTOGOTOURL parameter in the <OBJECT> tag to determine how URLs in the presentation are handled. The default value of true applies if you leave the parameter out. In this case any URL embedded in the presentation goes to the browser. If you set this parameter to false, RealPlayer sends the URL to the application with the OnGotoURL() call.

Note

The SetAutoGoToURL() method (see page 15) can override the AUTOGOTOURL parameter.

Using PREFETCH with <OBJECT>

The PREFETCH parameter causes RealPlayer G2 to get the stream description information before the stream starts. You can use this to find out the size and width of an embedded clip, for example, then dynamically create the <OBJECT> tag for the image window, using the clip's native size for the WIDTH and HEIGHT parameters. To do this, you would embed a control, such as the default control panel, and set PREFETCH to true:

```
<OBJECT ID=RVOCX CLASSID="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA"
WIDTH=400 HEIGHT=100>
<PARAM NAME="SRC" VALUE="rtsp://realserver.company.com/media/animate.swf">
<PARAM NAME="CONSOLE" VALUE="one">
<PARAM NAME="CONTROLS" VALUE="All">
<PARAM NAME="PREFETCH" VALUE="true">
</OBJECT>
```

The control fetches the stream information, then responds with the callback `OnPreFetchComplete()` (see page 31). You next use `GetClipWidth()` and `GetClipHeight()` (see page 18) to determine the native size of the image window. Through a means such as DHTML, you can then create the image window control. If the width and height were 320 by 240, respectively, you could generate an `<OBJECT>` tag like the following:

```
<OBJECT ID=RVOCX CLASSID="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA"
WIDTH=320 HEIGHT=240>
<PARAM NAME="SRC" VALUE="rtsp://realserver.company.com/media/animate.swf">
<PARAM NAME="CONSOLE" VALUE="one">
<PARAM NAME="CONTROLS" VALUE="ImageWindow">
</OBJECT>
```

Receiving Callbacks

To receive callbacks through VBScript, you use the `<OBJECT>` tag ID, shown here set to `RVOCX`:

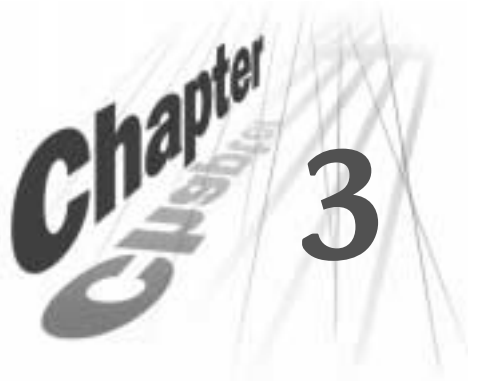
```
<OBJECT ID=RVOCX HEIGHT=256 WIDTH=256>
  CLASSID="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA"
<PARAM NAME="controls" VALUE="all">
<PARAM NAME="SRC" VALUE="http://www.company.com/sample.ram">
</OBJECT>
```

You then use a `<SCRIPT>` tag to receive a VBScript callback. The following example shows a callback for `onShowStatus(statusText)`:

```
<P>
Status Text:
<input type="text" name="statusText" size=100><br>
</P>
<SCRIPT language="VBS">
Sub RVOCX_OnShowStatus(byVal text)
  statusText.Value=text
End Sub
</SCRIPT>
```

Additional Information

The section "CallBack Methods" beginning on page 28 lists the RealPlayer callback events.



METHODS

An application, applet, or control can use the following methods to communicate with RealPlayer G2. For method data types, refer to RAPlayer.java, included in the rpcl3260.zip file installed in the RealPlayer folder. RealPlayer 5.0 supports a limited set of these methods as described in "RealPlayer 5.0 Compatibility" on page 27.

Playback Commands

DoPlay()
Plays the current clip. Equivalent to clicking the Play button.

CanPlay()
Returns TRUE or FALSE. Returns TRUE if the player is currently paused or stopped, and current source file is valid.

DoStop()
Stops the clip. Equivalent to clicking the Stop button. This method is backwards-compatible with RealPlayer 5.0.

CanStop()
Returns TRUE if RealPlayer is currently playing a clip or is paused. This method is backwards-compatible with RealPlayer 5.0.

DoPause()
Pauses the current clip. Equivalent to clicking the Pause button.

CanPause()
Returns TRUE if the player is currently playing a clip.

CanPlayPause()
Returns `TRUE` if the player is currently playing, paused, or stopped, and current source file is valid. Backwards-compatible with RealPlayer 5.0. If supporting only RealPlayer G2, use `CanPause()` and `CanPlay()` instead.

DoPlayPause()
Plays or pauses the current clip. Backwards-compatible with RealPlayer 5.0. If supporting only RealPlayer G2, use `DoPause()` and `DoPlay()` instead.

PlayState Information

GetPlayState()
Returns the current state of the RealPlayer. Values are numerical, with the following meanings:

```
0 Stopped
1 Contacting
2 Buffering
3 Playing
4 Paused
5 Seeking
```

GetBufferingTimeElapsed()
Returns number of milliseconds of elapsed buffering time.

> **Note**
> The `GetBufferingtimeElapsed()` method is not currently supported. It will be supported in an upcoming release.

GetBufferingTimeRemaining()
Returns estimated remaining buffering time in milliseconds.

Attributes

GetPreFetch()
Returns whether or not PreFetch is enabled.

SetPreFetch()
Enables/disables PreFetch playback mode. Valid values are `TRUE` and `FALSE` with `FALSE` as the default.

Additional Information

See "Using PREFETCH with <EMBED>" on page 5 or "Using PREFETCH with <OBJECT>" on page 9.

GetControls()

Returns the visible components of the control.

Additional Information

For valid control names, see the chapter on Web page playback in *RealSystem G2 Production Guide* at **http://service.real.com/help/library/index.html**.

SetControls()

Sets the visible components of the control. Compatible only with RealPlayer G2.

SetControlString()

Sets the visible components of the control. Method is identical to `SetControls()`, but is compatible with RealPlayer 5.0 Netscape Plug-in (though not the 5.0 ActiveX control). If supporting only RealPlayer G2, use `SetControls()` instead.

GetConsole()

Returns a console name used to link multiple control instances. Call this function once for each instance of a control you want to link.

SetConsole()

Sets a console name used to link multiple control instances. Call this once for each instance of a control you want to link. All controls with the same console name work together. For example, if you have multiple Play and Stop buttons on the same page, a shared console name enables them to control the same clip. The console name `_master` links to all instances. The console name `_unique` links to no other instances.

SetConsoleName()

Functionally identical to `SetConsole()`, but backwards-compatible with RealPlayer 5.0 Netscape plug-in (though not the 5.0 ActiveX control). If supporting only RealPlayer G2, use `SetConsole()` instead.

SetAutoStart(boolean autoStart)
Sets whether or not the control automatically starts playing once the source data is available. Valid values are `TRUE` and `FALSE`. This method is compatible with the RealPlayer 5.0 Netscape Plug-in but not its ActiveX control.

GetAutoStart()
Returns whether or not playback will start automatically.

SetAutoGoToURL()
Specifies how a URL will be handled. This method is backwards-compatible with RealPlayer 5.0. Valid values are:

- `TRUE`

  RealPlayer plug-in automatically forwards URL event to browser.

- `FALSE`

  onGoToURL() event handled by Java applet or VBScript instead.

  Additional Information
  With RealPlayer G2, this can also be set with the `AUTOGOTOURL` parameter in the `<EMBED>` or `<OBJECT>` tag. See "Handling URLs through <EMBED>" on page 5 or "Handling URLs through <OBJECT>" on page 9.

GetAutoGotoURL()
Returns whether or not AutoGotoURL property is enabled.

DoGoToURL(url, target)
For the RealPlayer 5.0 and G2 ActiveX control only, this method causes the control to attempt a navigation to the specified URL in the specified frame target. The container must support URL browsing. Parameters are:

`string URL`

`string target`

SetLoop()
Specifies whether the clip will loop or not. Valid values are `TRUE` (loops until play is interrupted) and `FALSE` (default).

GetLoop()
Returns whether the clip has been set to loop.

SetNumLoop()
Sets number of times to loop the clip. Takes an integer as its parameter.

GetNumLoop()
Returns number of loops set in `SetNumLoop()`.

SetCenter()
Sets whether or not the visual datatype should be centered at its natural size within the image window. Valid values are `TRUE` and `FALSE` (default).

GetCenter()
Returns whether or not the visual datatype will be centered within the image window. Valid values are `TRUE` and `FALSE`.

SetMaintainAspect()
Maintains correct aspect ratio of source within image window when stretched. Valid values are `TRUE` and `FALSE` (default).

GetMaintainAspect()
Returns whether or not the aspect ratio of the visual datatype will be maintained.

SetBackgroundColor()
Specifies the desired background color for the image window control. Valid values are a RGB hexadecimal color value in the format `#RRGGBB`, or the following color names, shown here with their corresponding RGB values:

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

GetBackgroundColor()
Returns hexadecimal value for current background color.

SetNoLogo()
Determines whether to suppress the display of RealLogo in image window control. Image window defaults to black unless a background color has been specified. Valid values are `TRUE` and `FALSE` (default).

GetNoLogo()
Returns whether or not the RealLogo will be displayed in the image window.

SetNoLabels()
Suppresses the Title, Author, and Copyright label text in the controls window of RealPlayer 5.0. The content text strings are still displayed. This method does not affect RealPlayer G2 but is compatible with it.

GetNoLabels()
Returns whether or not Title, Author, and Copyright labels will be suppressed.

SetShuffle()
Randomizes playback of all clips, excluding clips that have already played. Works for multiclip RAM files (`.ram` or `.rpm`) or SMIL files that contain only a sequence of clips. Valid values are `TRUE` and `FALSE`.

GetShuffle()
Returns whether shuffle play is enabled.

SetSource(String Source)
Specifies the URL of the clip to play. The source URL can begin with `rtsp://`, `http://`, `pnm://`, or `file://`. This method is backwards-compatible with the RealPlayer 5.0 Netscape plug-in, though not the 5.0 ActiveX control.

GetSource()
Returns the URL of the playing clip.

SetCanSeek()
Sets whether the user can seek within the clip through the user interface. Valid values are `TRUE` (default) and `FALSE`. Live or simulated live clips are always `FALSE`.

GetCanSeek()
Live or simulated live clips return `FALSE`.

Clip Position

GetPosition()
Returns the current position in the clip in milliseconds. Valid values will be `>=0` and `<=`*`total clip length`*.

SetPosition()
Seeks into the clip the specified point in milliseconds. Valid values are `=0` and `<`*`total clip length`*.

If an attempt is made to set the position >total length, then SetPosition() will equal total length.

GetLength()
Returns the total length of the clip in milliseconds. Valid values are >=0.

Clip/TAC Information

GetTitle()
Returns the current clip's title string.

SetTitle()
Sets the current clip's title string, overriding any existing title information. GetTitle() subsequently returns this new value.

GetAuthor()
Returns the current clip's author string.

SetAuthor()
Sets the current clip's author string, overriding any existing author information. GetAuthor() subsequently returns this new value.

GetCopyright()
Returns the current clip's copyright string.

SetCopyright()
Sets the current clip's copyright string, overriding any existing copyright information. GetCopyright() subsequently returns this new value.

GetClipWidth()
Returns the width of the clip window in pixels.

GetClipHeight()
Returns the height of the clip window in pixels.

Playlist/Multiclip

HasNextEntry()
Tests if the next clip function is available. The next clip function is available when the connected source is a RAM (.ram or .rpm) or SMIL file that contains

multiple clips and the current clip is not the last clip in the RAM or SMIL file. In a SMIL file, a <par> group is treated as a single clip. Returns TRUE or FALSE.

HasNextItem()
Tests if the next clip function is available. The next clip function is available when the connected source is a RAM (.ram or .rpm) or SMIL file that contains multiple clips and the current clip is not the last clip in the RAM or SMIL file. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use HasNextEntry() instead.

DoNextEntry()
Skips to the next clip in the RAM (.ram or .rpm) or SMIL file that contains multiple clips. In a SMIL file, a <par> group is treated as a single clip.

DoNextItem()
Skips to the next clip in the RAM (.ram or .rpm) or SMIL file that contains multiple clips. In a SMIL file, a <par> group is treated as a single clip. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use DoNextEntry() instead.

HasPrevEntry()
Tests if the previous clip function is available. The previous clip function is available when the connected source is a RAM (.ram or .rpm) or SMIL file that contains multiple clips and the current clip is not the first clip in the RAM file. In a SMIL file, a <par> group is treated as a single clip. Returns TRUE or FALSE.

HasPrevItem()
Tests if the previous clip function is available. The previous clip function is available when the connected source is a RAM (.ram or .rpm) or SMIL file that contains multiple clips and the current clip is not the first clip in the RAM file. In a SMIL file, a <par> group is treated as a single clip. Returns TRUE or FALSE. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use HasPrevEntry() instead.

DoPrevEntry()
Skips to the previous clip in a RAM (.ram or .rpm) or SMIL file that contains multiple clips. In a SMIL file, a <par> group is treated as a single clip.

DoPrevItem()
Skips to the previous clip in a RAM (`.ram` or `.rpm`) or SMIL file that contains multiple clips. In a SMIL file, a `<par>` group is treated as a single clip. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use `DoPrevEntry()` instead.

GetCurrentEntry()
Returns the number of the entry currently playing.

GetNumEntries()
Returns the total number of entries in the playlist.

GetEntryTitle()
Returns the Title for the specified playlist entry.

GetEntryAuthor()
Returns the Author for the specified playlist entry.

GetEntryCopyright()
Returns the Copyright for the specified playlist entry.

GetEntryAbstract()
Returns the Abstract for the specified playlist entry.

Live Playback

GetLiveState()
Returns whether the current clip is live. Valid values are `TRUE` or `FALSE`.

User Interface Components

SetShowStatistics()
Sets the RealPlayer Statistics dialog box to visible. Valid values are `TRUE` or `FALSE`.

GetShowStatistics()
Returns whether or not the RealPlayer Statistics dialog box is visible.

HideShowStatistics()
Sets the RealPlayer Statistics dialog box to visible. Valid values are `TRUE` (hidden) or `FALSE`. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use `SetShowStatistics()` instead.

IsStatisticsVisible()
Returns whether or not the RealPlayer Statistics dialog box is visible. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use `GetShowStatistics()` instead.

SetShowPreferences()
Displays the RealPlayer Preferences dialog box.

GetShowPreferences()
Returns whether or not the Preferences dialog box is visible.

EditPreferences()
Displays the RealPlayer Preferences dialog box. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use `SetShowPreferences()` instead.

SetShowAbout()
Displays the RealPlayer About dialog box. Use with RealPlayer G2 only.

AboutBox()
Displays the RealPlayer About dialog box. Use with RealPlayer G2 and 5.0. If supporting only RealPlayer G2, use `SetShowAbout()` instead.

GetShowAbout()
Returns whether or not the About box is open.

Error Handling

SetWantErrors()
Sets error sink. Valid values are `TRUE` (errors trapped, no error dialogs in player) and `FALSE` (error dialogs in player).

GetWantErrors()
Returns whether error dialogs will be displayed.

GetLastErrorMoreInfoURL()
Returns the "more info" URL from the last error. May return nothing.

GetLastErrorRMACode()

Returns RMA error code from the last error. RMA error codes are described in the header file `pnresult.h` in the RealSystem G2 SDK available at **http://www.real.com/devzone/**. In normal operation, all RealSystem components need to be able to handle the following basic codes that may be returned by the RealSystem system:

- PNR_FAIL—Operation failed.
- PNR_OK—Operation succeeded.
- PNR_UNEXPECTED—Call was unexpected or method is not implemented.

GetLastErrorSeverity()

Returns error level for last error.

Error Levels

Level	Condition	Usage
0	Panic	Error potentially causing a system failure. RealSystem takes actions necessary to correct the problem. This may include shutting down the presentation.
1	Severe	Error requiring immediate user intervention to prevent a problem. RealSystem will shut down the presentation if necessary.
2	Critical	Error that may require user intervention to correct. RealSystem will shut down the presentation if necessary.
3	General	Error that does not cause a significant problem with normal system operation.
4	Warning	Warning about a condition that does not cause system problems but may require attention.
5	Notice	Notice about a condition that does not cause system problems but should be noted.
6	Informational	Informational message only.
7	Debug	Information of use only when debugging a program.

GetLastErrorUserCode()

Returns user error code from last error.

GetLastErrorUserString()

Returns error string from last error dialog. May return nothing.

GetLastStatus()
Returns text of last status message.

Display State

SetOriginalSize()
Sets the image window to its original size.

GetOriginalSize()
Returns whether the image is currently in its original size. `TRUE` indicates original size.

SetDoubleSize()
Sets the image window to double size.

GetDoubleSize()
Returns whether or not image is currently in double-size mode. `TRUE` indicates double size.

SetFullScreen()
Sets the image to full-screen mode.

GetFullScreen()
Returns whether or not image is currently in full-screen mode. `TRUE` indicates full-screen mode.

Volume/Mute Control

SetVolume()
Sets the volume level. Valid values are 0-100.

GetVolume()
Returns current volume level.

SetMute()
Sets the mute state. Valid values are `TRUE` and `FALSE`.

GetMute()
Returns whether or not the volume has been muted.

GetStereoState()
Returns whether the current clip is in stereo. Returns TRUE for stereo or FALSE for mono.

Context Menu

SetEnableContextMenu()
Specifies whether the control will display the default context menu when the right mouse button is clicked. Valid values are TRUE (enabled, default) and FALSE (disabled).

GetEnableContextMenu()
Returns whether or not contextual menu is currently enabled. Returns TRUE (enabled, default) or FALSE (disabled).

SetEnableOriginalSize()
Enables the original size option on contextual menu. Valid values are TRUE (default) and FALSE.

GetEnableOriginalSize()
Returns whether the original size option is enabled on the contextual menu.

SetEnableDoubleSize()
Enables the double size option on contextual menu. Valid values are TRUE and FALSE (default).

GetEnableDoubleSize()
Returns whether or not the double size option is enabled on the contextual menu.

SetEnableFullScreen()
Enables the full screen option on contextual menu. Valid values are TRUE (default) and FALSE.

GetEnableFullScreen()
Returns whether or not the full screen option is enabled on the contextual menu.

SetImageStatus()
Sets whether Status text should be written to the image window.

GetImageStatus()
Returns whether Status text is written to the image window.

Network Information

GetSourceTransport()
Returns a string with the source protocol used for playback. Takes as value an integer from 1 to *n*, where *n* is the number of sources returned by `GetNumSources()`.

GetNumSources()
Returns number of sources in the presentation.

GetPacketsTotal()
Returns total number of packets in the presentation.

GetPacketsReceived()
Returns the total packets received from the server.

GetPacketsOutofOrder()
Returns the total packets received from the server out of order.

GetPacketsMissing()
Returns the total packets not received from the server in time to play.

GetPacketsEarly()
Returns the total packets received from the server before they are ready to play.

GetPacketsLate()
Returns the total packets received from the server that are too late to play.

GetBandwidthAverage()
Returns the average amount of bandwidth available.

GetBandwidthCurrent()
Returns the current amount of bandwidth available.

Version Information

GetVersionInfo()
Returns major and minor version information for the embedded RealPlayer (not the parent RealPlayer), such as `6.0.0.128`.

GetIsPlus()
Returns whether the client is RealPlayer Plus.

Event Flags

SetWantKeyboardEvents()
Sets whether keyboard events will be sent. Valid values are `TRUE` and `FALSE` (default).

GetWantKeyboardEvents()
Returns whether keyboard events will be sent.

SetWantMouseEvents()
Sets whether mouse events will be sent. Valid values are `TRUE` and `FALSE` (default).

GetWantMouseEvents()
Returns whether mouse events will be sent.

User Preference Settings

GetConnectionBandwidth()
Returns the normal, maximum bandwidths settings as set by the user in the RealPlayer preferences.

GetPreferedLanguageString()
Returns the preferred language for content as set by the user in the RealPlayer preferences.

GetPreferedLanguageID()
Returns the preferred language ID. For a list of language codes, see the SMIL language codes appendix in *RealSystem G2 Production Guide*.

GetUserCountryID()
Returns the country that the user selected during electronic registration.

RealPlayer 5.0 Compatibility

The following methods are used with RealPlayer G2 and RealPlayer 5.0. To support both versions of RealPlayer, your application can issue only this set of commands:

- DoStop() on page 12
- CanStop() on page 12
- CanPlayPause() on page 13
- DoPlayPause() on page 13
- DoGoToURL(url, target) on page 15 (ActiveX control only)
- SetControlString() on page 14 (Netscape Plug-in only)
- SetConsoleName() on page 14 (Netscape Plug-in only)
- SetAutoStart(boolean autoStart) on page 15 (Netscape Plug-in only)
- SetAutoGoToURL() on page 15
- SetNoLabels() on page 17 (Netscape Plug-in only)
- SetSource(String Source) on page 17 (Netscape Plug-in only)
- HasNextItem() on page 19
- DoPrevItem() on page 20
- HasPrevItem() on page 19
- DoPrevItem() on page 20
- AboutBox() on page 21
- EditPreferences() on page 21
- HideShowStatistics() on page 21
- IsStatisticsVisible() on page 21

CALLBACK METHODS

This chapter describes the methods RealPlayer calls to inform an application of its state.

RealPlayer G2 Callbacks

onClipOpened(shortClipName,URL)
Sent when a clip is opened by the control. Parameters:

`string shortClipName`

`string URL`

onClipClosed()
Sent to indicate that no clip is currently opened by the control. No parameters.

onShowStatus(statusText)
Sent to indicate that the status text is changing. Parameter:

`string statusText`

onGoToURL(URL,target)
Sent when an URL event is encountered for the RealPlayer clip currently playing. This event occurs only if the **AutoGotoURL** property is `FALSE`. Parameters:

`string URL`

`string target`

OnPositionChange(current position, total length)
Called when the position in the clip changes. The current position of the clip (in milliseconds) is returned. Parameter:

`long position`

`long length`

OnLengthChange(length)
Called when the length of the clip changes, generally when a new clip is loaded. Parameter:

`long length (in milliseconds)`

OnVolumeChange(volume)
Called when the volume level changes. The current volume level value is returned. The valid volume range is 0-100, where 0 represents no volume. Parameter:

`short volume`

OnTitleChange(title)
Called when the title string changes. The new title string is returned. Parameter:

`string title`

OnAuthorChange(author)
Called when the author string changes. The new author string is returned. Parameters:

`string author`

OnCopyrightChange(copyright)
Called when the copyright string changes. The new copyright string is returned. Parameters:

`string copyright`

OnPlayStateChange(newPlayState)
Called when the play state changes. See `GetPlayState()` on page 13 for play state values. Parameter:

`long newPlayState`

OnErrorMessage(Severity, RMAErrorCode, UserErrorCode, messageText, MoreInfoURL)
Called when an error occurs. See `GetLastErrorRMACode()` and `GetLastErrorSeverity()` starting on page 22 for more information on error codes and severities.

OnBuffering(int Flags, short PercentComplete)
Returns percentage of buffering complete.

Note

The `OnBuffering()` method is not currently supported. It will be supported in an upcoming release.

OnContacting(String Hostname)

Called when RealPlayer contacts a host. Returns host name string.

OnKeyDown(int KeyCode)

Returns key code when user presses and holds down a keyboard key. Sent only when `SetWantKeyboardEvents()` (see page 26) is set to `TRUE`.

OnKeyPress(int KeyCode)

Returns key code when user presses and releases a keyboard key. Sent only when `SetWantKeyboardEvents()` (see page 26) is set to `TRUE`.

OnKeyUp(int Keycode)

Returns key code when user releases keyboard key. Sent only when `SetWantKeyboardEvents()` (see page 26) is set to `TRUE`.

OnLButtonDown(int nButtonFlags, int xPos, int yPos)

Returns mouse position and flags when user holds down the left mouse button. Sent only when `SetWantMouseEvents()` (see page 26) is set to `TRUE`.

OnLButtonUp(int nButtonFlags, int xPos, int yPos)

Returns mouse position and flags when user releases the left mouse button. Sent only when `SetWantMouseEvents()` (see page 26) is set to `TRUE`.

OnRButtonDblClk(int nButtonFlags, int xPos, int yPos)

Returns mouse position and flags when user double clicks the right mouse button. Sent only when `SetWantMouseEvents()` (see page 26) is set to `TRUE`.

OnRButtonUp(int nButtonFlags, int xPos, int yPos)

Returns mouse position and flags when user releases the right mouse button. Sent only when `SetWantMouseEvents()` (see page 26) is set to `TRUE`.

OnMouseMove(int nButtonFlags, int xPos, int yPos)

Returns mouse position and flags when user moves the mouse. Sent only when `SetWantMouseEvents()` (see page 26) is set to `TRUE`.

OnMute(boolean bIsMute)

Called when volume is muted.

OnPreFetchComplete()
Called when component has fetched the stream header information. Called if `PREFETCH` is set to true in the `<EMBED>` or `<OBJECT>` tag or `GetPreFetch()` (see page 13) is set to `TRUE`.

OnPreSeek(int OldTime, int NewTime)
Called when the user performs a seek by moving the presentation position slider. Returned values include the presentation time when the seek occured and the value for the seek-to time.

OnPostSeek(int OldTime, int NewTime)
Called when a seek completes. Returned values include the presentation time before the seek occured and the time value after the seek occurs.

OnPresentationClosed()
Called when the presentation stops.

OnPresentationOpened()
Called when the presentation starts.

RealPlayer 5.0 Compatibility

The following callback methods are used with RealPlayer G2 and RealPlayer 5.0. If you support both versions of RealPlayer, your application will receive only the following:

- onClipOpened(shortClipName,URL) on page 28
- onClipClosed() on page 28
- onShowStatus(statusText) on page 28
- onGoToURL(URL,target) on page 28

METHOD INDEX

A AboutBox(), 21

C CanPause(), 12
CanPlay(), 12
CanPlayPause(), 13
CanStop(), 12

D DoGoToURL(), 15
DoNextEntry(), 19
DoNextItem(), 19, 20
DoPause(), 12
DoPlay(), 12
DoPlayPause(), 13
DoPrevEntry(), 19
DoStop(), 12

E EditPreferences(), 21

G GetAuthor(), 18
GetAutoGotoURL(), 15
GetAutoStart(, 15
GetBackgroundColor(), 16
GetBandwidthAverage(), 25
GetBandwidthCurrent(), 25
GetBufferingTimeElapsed(), 13
GetBufferingTimeRemaining(), 13
GetCanSeek(), 17
GetCenter(), 16
GetClipHeight(), 18
GetClipWidth(), 18
GetConnectionBandwidth(), 26
GetConsole(), 14
GetControls(), 14
GetCopyright(), 18
GetCurrentEntry(), 20
GetDoubleSize(), 23
GetEnableContextMenu(), 24
GetEnableDoubleSize(), 24
GetEnableFullScreen(), 24
GetEnableOriginalSize(), 24
GetEntryAbstract(), 20
GetEntryAuthor(), 20
GetEntryCopyright(), 20
GetEntryTitle(), 20
GetFullScreen(), 23
GetImageStatus(), 25
GetIsPlus(), 26
GetLastErrorMoreInfoURL(), 21
GetLastErrorRMACode(), 22
GetLastErrorSeverity(), 22
GetLastErrorUserCode(), 22
GetLastErrorUserString(), 22
GetLastStatus(), 23
GetLength(), 18
GetLiveState(), 20
GetLoop(), 15
GetMaintainAspect(), 16
GetMute(), 23
GetNoLabels(), 17
GetNoLogo(), 16
GetNumEntries(), 20
GetNumLoop(), 16
GetNumSources(), 25
GetOriginalSize(), 23
GetPacketsEarly(), 25
GetPacketsLate(), 25
GetPacketsMissing(), 25
GetPacketsOutofOrder(), 25

GetPacketsReceived(), 25
GetPacketsTotal(), 25
GetPlayState(), 13
GetPosition(), 17
GetPreferedLanguageID(), 26
GetPreferedLanguageString(), 26
GetPreFetch(), 13
GetShowAbout(), 21
GetShowPreferences(), 21
GetShowStatistics(), 20
GetShuffle(), 17
GetSource(), 17
GetSourceTransport(), 25
GetStereoState(), 24
GetTitle(), 18
GetUserCountryID(), 27
GetVersionInfo(), 26
GetVolume(), 23
GetWantErrors(), 21
GetWantKeyboardEvents(), 26
GetWantMouseEvents(), 26

H

HasNextEntry(), 18
HasNextItem(), 19
HasPrevEntry(), 19
HasPrevItem(), 19
HideShowStatistics(), 21

I

IsStatisticsVisible(), 21

O

OnAuthorChange(), 29
OnBuffering(), 29
onClipClosed(), 28
onClipOpened(), 28
OnContacting(), 30
OnCopyrightChange(), 29
OnErrorMessage(), 29
onGoToURL(), 28
OnKeyDown(), 30
OnKeyPress(), 30
OnKeyUp(), 30
OnLButtonDown(), 30
OnLButtonUp(), 30
OnLengthChange(), 29
OnMouseMove(), 30
OnMute(), 30
OnPlayStateChange(), 29
OnPositionChange(), 28
OnPostSeek(), 31
OnPreFetchComplete(), 31
OnPreSeek(), 31
OnPresentationClosed(), 31
OnPresentationOpened(), 31
OnRButtonDblClk(), 30
OnRButtonUp(), 30
onShowStatus(), 28
OnTitleChange(), 29
OnVolumeChange(), 29

S

SetAuthor(), 18
SetAutoGoToURL(), 15
SetAutoStart(), 15
SetBackgroundColor(), 16
SetCanSeek(), 17
SetCenter(), 16
SetConsole(), 14
SetConsoleName(), 14
SetControls(), 14
SetControlString(), 14
SetCopyright(), 18
SetDoubleSize(), 23
SetEnableContextMenu(), 24
SetEnableDoubleSize(), 24
SetEnableFullScreen(), 24
SetEnableOriginalSize(), 24
SetFullScreen(), 23
SetImageStatus(), 24
SetLoop(), 15
SetMaintainAspect(), 16
SetMute(), 23
SetNoLabels(), 17
SetNoLogo(), 16
SetNumLoop(), 16
SetOriginalSize(), 23

SetPosition(), 17
SetPreFetch(), 13
SetShowAbout(), 21
SetShowPreferences(), 21
SetShowStatistics(), 20
SetShuffle(), 17
SetSource(), 17
SetTitle(), 18
SetVolume(), 23
SetWantErrors(), 21
SetWantKeyboardEvents(), 26
SetWantMouseEvents(), 26