



REAL VISUALIZATION SOFTWARE DEVELOPMENT KIT DEVELOPER'S GUIDE

Version 2
August 2000



Information in this document is subject to change without notice. Companies, names, and data used in examples herein are fictitious unless otherwise noted. 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.

© 2000 RealNetworks, Inc.

RealAudio, RealVideo, and RealPlayer are registered trademarks of RealNetworks, Inc.

The Real logo, RealServer, RealPlayer Plus, RealText, RealPix, RealAudio Encoder, RealVideo Encoder, RealEncoder, RealPublisher, RealProducer, RealProducer Plus, RealProducer Pro, SureStream, RealBroadcast Network, RealJukebox, and RealSystem are trademarks of RealNetworks, Inc.

Microsoft, MS-DOS, Windows, and Windows NT are registered trademarks of Microsoft Corporation.

Other product and corporate names may be trademarks or registered trademarks of other companies. They are used for explanation only, with no intent to infringe.

RealNetworks, Inc.
2601 Elliott Avenue
Seattle, WA 98121 USA

<http://www.real.com>



CONTENTS

	INTRODUCTION	5
	Overview	5
	What's New in Version 2 of the Real Visualization SDK.....	5
	New APIs	5
	New Samples	6
	How This Manual Is Organized	6
	Text Conventions in This Manual	6
	Technical Support	7
1	USING THE SDK	9
	Introduction	9
	Contents of the SDK.....	9
	SDK Header Files	11
	Sample Files.....	11
2	DEVELOPMENT FRAMEWORK	13
	How the Real Visualization SDK Differs from COM	13
	Creating a Plug-In Instance	13
	Creating a RealSystem Object	14
	Using AddRef() and Release()	14
	Suggestions for Designing a Plug-In	15
	Compiling a Plug-In.....	15
	Building the Sample Plug-In	15
	Debugging a Plug-In	16
	Testing a Plug-In	16
3	THE VISUALIZATION INTERFACE	17
	Using the Visualization Interface	17
	Buffer Mode Rendering.....	17
	Window Mode Rendering.....	18
A	INTERFACE LIST	21
	IMPVisualization	21
	IMPVisualizationPlugin::InitPlugin	22
	IMPVisualizationPlugin::GetPluginInfo	22

	IMPAVisualizationPlugin::GetPluginProperties.....	22
	IMPAVisualizationPlugin::SetPreferences	23
	IMPAVisualizationPlugin::Configure.....	23
	IMPAVisualizationPlugin::SetRender	24
	IMPAVisualizationPlugin::OnDrawBuffer.....	24
	IMPAVisualizationPlugin::OnClick	25
	IMPAVisualizationPlugin::SetWindow	25
	IMPAVisualizationPlugin::OnDrawWindow	26
<i>B</i>	LICENSE AGREEMENT	27



INTRODUCTION

Welcome to the Real Visualization Software Development Kit (SDK)!

Overview

RealNetworks® has created this SDK for developers working with the RealJukebox™ and RealPlayer® applications. This manual helps you use the SDK to produce visualization plug-ins for RealJukebox and RealPlayer. A visualization plug-in is a DLL that renders a graphical display while audio is being played by RealJukebox or RealPlayer. The graphic rendering is synchronized to the amplitude of the audio to provide a lively visual effect. After you build a visualization plug-in, submit your DLL to RealNetworks at **upload_vis@real.com** for review and distribution on our RealJukebox Central web site.

Additional Information

Be sure to read the SDK license agreement in full. See “Appendix B: License Agreement” on page 27.

What’s New in Version 2 of the Real Visualization SDK

New APIs

- Window support. Your visualization plug-in can use either window mode rendering or buffer mode rendering.
 - In window mode, your plug-in is handed a window with the SetWindow function. Rendering status information is supplied to your plug-in with the SetRender function. Finally, your plug-in is given audio data with the OnDrawWindow function.

- In buffer mode, the visualization graphic is rendered in a buffer that is then rendered by the application, either RealJukebox or RealPlayer, in the visualization window pane.
- Configuration support. Your visualization plug-in can create plug-in-specific configuration data to save state information. This new configuration support allows you to create a modal dialog box where you can specify attribute settings associated with your visualization plug-in. Because these settings are persistent, they are used every time the visualization plug-in runs.

New Samples

Version 2 of the Real Visualization SDK includes two sample visualizations.

- The first sample shows how to build a visualization by drawing into buffers.
- The second sample demonstrates the new window support mode.

How This Manual Is Organized

This manual contains the following information:

- “Chapter 1: Using the SDK” on page 9 describes the files included in the Real Visualization SDK.
- “Chapter 2: Development Framework” on page 13 discusses the Real Visualization SDK’s implementation of COM, and lists the steps for compiling the sample plug-ins provided with the SDK.
- “Chapter 3: The Visualization Interface” on page 17 explains the methods used for buffer mode and window mode rendering.
- “Appendix A: Interface List” on page 21 describes the IMPAVisualization interface and its methods.
- “Appendix B: License Agreement” on page 27 contains the SDK license agreement.

Text Conventions in This Manual

Interfaces, methods, functions, code excerpts, and file names appear in a sans serif typeface.

User interface elements appear in **Bold Title Case**.

Technical Support

For technical support for the Real Visualization SDK, e-mail:

supportsdk@real.com

For general information about RealNetworks' technical support, visit:

<http://service.real.com>

USING THE SDK

This chapter describes the header files and sample files that comprise the Real Visualization SDK.

Introduction

RealSystem™ is based on the Component Object Model (COM) binary standard jointly developed by Microsoft Corporation and Digital Equipment Corporation. Because RealSystem is based on COM, you can develop RealSystem components using virtually any programming language. However, if you want to use the sample files provided with the SDK, you must use C++. You should be familiar with COM before you begin developing RealSystem components, such as visualization DLLs.

Note

RealSystem diverges from the COM standard to simplify cross-platform development.

Additional Information

See “Chapter 2: Development Framework” on page 13 for information on how RealSystem implements COM.

Contents of the SDK

The following table shows the files included in the Real Visualization SDK.

Files in the Real Visualization SDK	
File name	Description
RealVisSDKv2.pdf	This document

(Table Page 1 of 3)

Files in the Real Visualization SDK (continued)

File name	Description
BufferMode directory	Sample source code for buffer mode
SampleVisualPlugin.dsp	Sample plug-in project file
SampleVisualPlugin.rc	Resource file
SampleVisualPlugin.rc2	Resource file
SampleVisualPluginBufferMode.cpp	Sample visualization using buffer mode
SampleVisualPluginBufferMode.h	Header file for sample plug-in
SampleVisualPlugin.dsw	Workspace file
pncom.h	Header file
pnbastsd.h	Header file
guids.cpp	GUID header file
resource.h	Resource header file
impavisualization.h	Interface header file
pnresult.h	Error codes
pntypes.h	Data types
pnwintyp.h	Window types
rmmacomm.h	Used for configuration settings
rmapckts.h	Used for configuration settings
rmavalue.h	Used for configuration settings
rmaprefs.h	Used for configuration settings
WindowMode directory	Sample source code for window mode
SampleVisualPlugin.dsp	Sample plug-in project file
SampleVisualPlugin.rc	Resource file
SampleVisualPlugin.rc2	Resource file
SampleVisualPluginWindowMode.cpp	Sample visualization using window mode
SampleVisualPluginWindowMode.h	Header file for sample plug-in
SampleVisualPlugin.dsw	Workspace file
pncom.h	Header file
pnbastsd.h	Header file
guids.cpp	GUID header file

(Table Page 2 of 3)

Files in the Real Visualization SDK (continued)

File name	Description
resource.h	Resource header file
impavisualization.h	Interface header file
pnresult.h	Error codes
pntypes.h	Data types
pnwintyp.h	Window types
rmacomm.h	Used for configuration settings
rmapckts.h	Used for configuration settings
rmavalue.h	Used for configuration settings
rmaprefs.h	Used for configuration settings

(Table Page 3 of 3)

SDK Header Files

The header files in this SDK define the RealSystem interfaces used to create a visualization plug-in. These files contain information that is not listed in the documentation, for example, function variables and return values. Consult both the header files and the documentation when you develop a visualization plug-in.

Sample Files

The BufferMode and WindowMode directories contain sample C++ files, as well as the header and project files required for making visualization plug-ins. If you use the sample files to learn how to build your own visualization plug-ins, you must know how to use C++.

Additional Information

RealNetworks recommends using specific compilers for compiling code based on the sample files. See “Compiling a Plug-In” on page 15 for more information.

This chapter explains how the Real Visualization SDK implements COM, and provides instructions for compiling the sample plug-in that is included with the SDK.

How the Real Visualization SDK Differs from COM

RealSystem components use the COM `QueryInterface()` method to expose their interfaces. The Real Visualization COM-style interfaces begin with the prefix `IMPA`, which stands for Interface Music Platform Architecture.

Although the Real Visualization SDK architecture is based on the COM standard, the SDK does not use all aspects of COM. Instead, it implements a subset of COM functions to provide cross-platform operation without requiring Windows libraries or Windows emulation code on UNIX and Macintosh platforms. The Real Visualization SDK thus eliminates the need for storage-intensive Windows components such as the registry and the COM and OLE runtime libraries. The following sections describe how the Real Visualization SDK diverges from the COM standard.

Additional Information

Visit <http://www.microsoft.com/com/default.asp> for more information on COM.

Creating a Plug-In Instance

The Real Visualization SDK does not use the Windows `CoCreateInstance()` method to create plug-in objects. Instead, each visualization plug-in implements `MPACreateVisualization()`, a C-style entry. The following extract from a Real Visualization SDK sample file illustrates the entry point.

```
/* Exported Entry Point - Create Function */
PN_RESULT DLLEXP MPACreateVisualization(IUnknown** ppIUnknown)
{
    // Create a new instance of a visualization plug-in...
```

```

    *ppIUnknown = (IUnknown*)new CSampleVisualPlugin();
    if (*ppIUnknown)
    {
        (*ppIUnknown)->AddRef();
        return PNR_OK;
    }
    // Out of memory
    return PNR_OUTOFMEMORY;
}

```

Creating a RealSystem Object

The Real Visualization SDK uses RealSystem interfaces. RealSystem COM-style interfaces begin with the prefix IMPA, which stands for Interface Music Platform Architecture, or IRMA, which stands for Interface RealMedia Architecture. A RealSystem component can use the C++ new operator to create objects that it alone manipulates. However, to create objects passed to other RealSystem components, a component should use IRMACommonClassFactory. When RealSystem initializes a component, it passes to the component a pointer to the system context. The component can then use this pointer to call IRMACommonClassFactory::CreateInstance() and create new RealSystem objects. The following extract from a RealSystem SDK sample file illustrates the function call.

```
m_pClassFactory->CreateInstance(CLSID_IRMABuffer, (void**)&pStringObj);
```

Using AddRef() and Release()

Because RealSystem objects are often utilized by other RealSystem objects, objects must correctly implement reference counting. The COM functions AddRef() and Release() control reference counting and therefore determine the lifetime of each object. The following rules describe the requirements for RealSystem components' use of AddRef() and Release() with objects.

- The following functions perform an AddRef() on objects before returning them. When a component finishes using the object, it must Release() the object.
 - MPACreateVisualization()
 - IUnknown::QueryInterface()

- If a component receives an object as a parameter of a function call, the component must perform an `AddRef()` on the object. When the component finishes using the object, it must `Release()` the object.
- If a component uses the C++ `new` operator to create an object, the component must perform an `AddRef()` on the object and then `Release()` the object when finished.

Suggestions for Designing a Plug-In

Keep the following general suggestions in mind as you develop plug-ins.

- Keep your plug-ins as simple, self-contained, and universal as possible.
- Consider the impact of the plug-in on other functions in `RealJukebox` and `RealPlayer`.

Compiling a Plug-In

The Real Visualization SDK includes two sample plug-ins that you can use as a basis for creating plug-ins.

- The `BufferMode` sample source code demonstrates how to create a visualization plug-in that renders the graphic in a buffer that is then rendered by the application, either `RealJukebox` or `RealPlayer`, in the visualization window pane.
- The `WindowMode` sample source code demonstrates how to render graphical data directly into a window pane.

Building the Sample Plug-In

Before you build your plug-in, you should test-compile the sample files. On Windows, RealNetworks recommends using Microsoft Visual C++ 4.2 or 6.0.

Additional Information

Visit <http://www.microsoft.com/products> for more information on Microsoft Visual C++.

A Microsoft Developer Studio project file is provided for the sample source code.

► To build the sample plug-in:

1. Using Microsoft Visual C++ 4.2 or later, open the included project file, `SampleVisualPlugin.dsp`.
2. Build the `SampleVisualPlugin.rpv` file.

Note

The `SampleVisualPlugin.rpv` file is copied to the `C:\Program Files\Common Files\Real\Visualizations` folder. The `SampleVisualPlugin.rpv` file must reside in this folder.

3. If RealJukebox is running, right-click on the visualization pane. **Sample Visual Plugin** should be listed in the visualization list.
4. Play some music and select **Sample Visual Plugin** from the visualization list.

Congratulations! You have successfully built and installed a visualization plug-in.

Debugging a Plug-In

You should be able to use your standard debugger to debug visualization plug-ins. In Visual C++, you can set the debugging settings in your DLL's project to use `realjukebox.exe` as the application for debugging.

Testing a Plug-In

Perform as much real-world testing of your plug-ins as possible. Be sure to test under conditions that are less than ideal. For example, try running your visualization while ripping a CD.

Chapter 3

THE VISUALIZATION INTERFACE

This chapter explains how RealJukebox and a visualization plug-in use the visualization interface. The sample code included with this SDK illustrates many of the features described here. You can use the sample code as a starting point for building your own plug-in.

Using the Visualization Interface

Every visualization plug-in implements the `IMPAVisualizationPlugin` interface (as defined in `impavisualization.h`). This interface creates the visualization.

Before you begin, make copies of the sample files included in this kit.

► To create a visualization plug-in:

1. Update the `SampleVisualizationPlugin.rc2` file, which contains the version information for your plug-in.
2. Be sure you export the `MPACreateVisualization` function.
3. Name the DLL with the `.rpv` extension.
4. Copy the `.rpv` file to the `C:\Program Files\Common Files\Real\Visualizations` folder. Because RealJukebox regularly checks this directory, you can copy the file to this directory while RealJukebox is running.

Buffer Mode Rendering

Buffer mode rendering uses the following methods:

- `CSampleVisualPlugin::Clicked`. Add code if you want mouse-click support, which notifies you when the visualization is being clicked on and is reacting to the click. In some modes in the RealJukebox application, you may not receive clicks when your visualization is being shown.
- `CSampleVisualPlugin::OnDrawBuffer`. Add your plug-in rendering code here.

Note

A 32-bit RGB video pointer is provided with the API. The API is set up for possible future support of additional color formats, depending on the screen resolution or other conditions. The video pointer points to the beginning of the buffer. In bottom-up DIBs, the first row is actually near the end of the buffer. Width and height are in pixels. Pitch is in number of bytes per row, and is negative because this is a bottom-up DIB.

You can render into a different buffer, such as a DC using Windows or OpenGL functions, and then copy the bits back into the passed-in buffer if you want to use different APIs than those used for direct rendering. However, be aware that the additional memcpy operation can be detrimental to visualization performance.

Window Mode Rendering

Window mode rendering uses the following methods:

- `CSampleVisualPlugin::GetPluginProperties`. To set up window mode, set the extended parameter `VIZ_RENDER_MODE` to the setting `VIZ_RENDER_WINDOW` in the `GetPluginProperties` function.
- `CSampleVisualPlugin::SetWindow`. When the visualization window is set, you are called with `SetWindow`. You can subclass the window, but you must remember to unsubclass the window when `SetWindow` is called with a different window handle or with a `NULL` window handle. `SetWindow` is called with a `NULL` window handle when your plug-in is supposed to get rid of the windows and is going away. When `RealJukebox` changes from docked to undocked visualizations, `SetWindow` is called with a different window. You may be better off creating a child window and using `SetParent` when you get different windows from `RealJukebox`, instead of subclassing and handling the painting as shown in the example in this document.
- `CSampleVisualPlugin::OnDrawWindow`. `OnDrawWindow` is called with the window handle, which should be the same handle you received in `SetWindow`, and the audio data. You should do all data processing on the audio data in this function call, not on your window thread. You should perform as much window manipulation as possible on the window thread

by sending or posting messages to the window, rather than in response to `OnDrawWindow` calls directly. `OnDrawWindow` is not called from the same thread that creates the window, which is the thread where your `WindowProc` is called.



INTERFACE LIST

IMPAVisualization

Purpose	Used to create visualization component for RealJukebox and RealPlayer
Header file	impavisualization.h

The IMPAVisualization interface contains the following methods:

IMPAVisualizationPlugin::InitPlugin
IMPAVisualizationPlugin::GetPluginInfo
IMPAVisualizationPlugin::GetPluginProperties
IMPAVisualizationPlugin::SetPreferences
IMPAVisualizationPlugin::Configure
IMPAVisualizationPlugin::SetRender
IMPAVisualizationPlugin::OnDrawBuffer
IMPAVisualizationPlugin::OnClick
IMPAVisualizationPlugin::SetWindow
IMPAVisualizationPlugin::OnDrawWindow

As with all COM interfaces, the IMPAVisualization interface inherits the following IUnknown methods:

IUnknown::AddRef
IUnknown::QueryInterface
IUnknown::Release

IMPAVisualizationPlugin::InitPlugin

Called with a context that the plug-in can use to perform a `QueryInterface()` for interfaces supported by the application.

```
STDMETHOD(InitPlugin) (  
    THIS_  
    IUnknown* pContext  
) PURE;
```

pContext
Pointer to the context.

IMPAVisualizationPlugin::GetPluginInfo

Called to get name, creator, and description information from the plug-in. All strings returned must contain no more than 256 characters.

```
STDMETHOD(GetPluginInfo) (  
    THIS_  
    char* szName, char* szCreator, char* szDescription  
) PURE;
```

szName
The name of the plug-in.

szCreator
Author of the plug-in.

szDescription
Description of the plug-in.

IMPAVisualizationPlugin::GetPluginProperties

Called to get information about the visualization effects supported and whether the plug-in wants spectral audio data. The `IRMAValues` `pExtendedParams` passed in are used for extending the properties that can be supported.

The extended parameters are:

- `VIZ_RENDER_MODE`, which defines the rendering mode the plug-in uses. Valid values are `VIZ_RENDER_BUFFER` and `VIZ_RENDER_WINDOW`. The default value is `VIZ_RENDER_BUFFER`.

- **VIZ_SUPPORT_CONFIGURE**, which the plug-in uses when it supports the new configuration feature. Valid values are 0 (false) or 1(true). The default value is 0.

```
STDMETHOD(GetPluginProperties) (
    THIS_ UINT32* pnEffectsFlags, bool* pbWantSpectrum,
    IRMAValues* pExtendedParams)
) PURE;
```

pnEffectsFlags

Lists the visualization effects that are supported.

pbWantSpectrum

Tells whether the plug-in should receive spectral audio data.

pExtendedParams

Lists the plug-in properties that are supported.

IMPVisualizationPlugin::SetPreferences

Passes a preferences interface to the plug-in. The plug-in can use this preferences interface to store its own settings. Use `ReadPref` and `WritePref` to get and set settings for your plug-in.

```
STDMETHOD(SetPreferences) (
    THIS_ IRMAPreferences* pPrefs)
) PURE;
```

pPrefs

Pointer to the preferences interface.

IMPVisualizationPlugin::Configure

Allows the plug-in to display its own configuration dialog if `SupportsConfigure` has been returned. You can cast `pParent->window` to an `HWND` to use as your parent window. Your dialog should be modal.

```
STDMETHOD(Configure) (
    THIS_ PNxWindow* pParent)
) PURE;
```

pParent

Pointer to the parent window.

IMPAVisualizationPlugin::SetRender

Tells the plug-in that it is going to be actively used for rendering, that is, that music is playing.

```
STDMETHOD(SetRender) (  
    THIS_ BOOL bRender  
) PURE;
```

bRender

Tells the plug-in whether it is going to be actively used for rendering.

When SetRender is called with bRender set to false, the window should be cleared, or reset to a default image if in VIZ_RENDER_WINDOW mode. The initial render mode should always be false.

IMPAVisualizationPlugin::OnDrawBuffer

Tells the plug-in to draw into the passed-in video buffer if the render mode is set to VIZ_RENDER_BUFFER.

```
STDMETHOD(OnDrawBuffer) (  
    THIS_ unsigned char* pVideoBuffer, int width, int height,  
    int pitch, int bitsperpixel, MPAVizAudioData* pAudioData  
) PURE;
```

pVideoBuffer

Points to the beginning of the buffer.

width

Width, in pixels, of the visualization.

height

Height, in pixels, of the visualization.

pitch

Pitch, in bytes per row, of the visualization. A negative value for pitch means that this is a bottom-up DIB, with the origin in the lower-left corner. Currently, pitch is always negative.

bitsperpixel

Currently, the value of bitsperpixel is always 32. Do not draw outside the lines. Possible future support for extended parameters may allow the plug-in to indicate support for more RGB formats, for example, 24-bit or 16-bit color.

pAudioData

Pointer to audio data.

IMPAVisualizationPlugin::OnClick

Informs the plug-in that a click has occurred on the plug-in.

```
STDMETHOD(OnClick) (  
    THIS_ int x, int y, int keys  
) PURE;
```

x

x-coordinate of the location of the click.

y

y-coordinate of the location of the click.

keys

Any combination of the following values defined by Windows:

- MK_Control (CTRL key is down)
- MK_RBUTTON (right mouse button is down)
- MK_SHIFT (SHIFT key is down)

IMPAVisualizationPlugin::SetWindow

Gives a window handle to the plug-in. The user can subclass the window handle to receive messages on size or hiding and showing. Any window created to handle drawing should not draw outside the area of this window, and this window should be the parent in order to maintain z-order.

The benefit of creating a child window is that when the visualization engine passes a new window to the plug-in, you can reset the parent of the window. For example, assume that the passed-in window is valid until SetWindow is called, where pWindow->window points to a different window, or to NULL. (NULL means no window.) When SetWindow is called with a new window, be sure to unsubclass the previous window passed in, and remove any child windows you previously added to that window.

Note

The only value you should use in the PNxWindow structure is pWindow->window. On Windows, you can cast that void* to HWND.

```
STDMETHOD(SetWindow) (  
    THIS_ PNxWindow* pWindow  
) PURE;
```

pWindow

Pointer to the window handle.

IMPAVisualizationPlugin::OnDrawWindow

Tells the plug-in to draw into its window if the render mode is set to VIZ_RENDER_WINDOW.

Note

The only value you should use in the PNxWindow structure is pWindow->window. On Windows, you can cast that void* to HWND.

```
STDMETHOD(OnDrawWindow) (  
    THIS_ PNxWindow* pWindow, MPAVizAudioData* pAudioData  
) PURE;
```

pWindow

Pointer to the window that was last passed in to the SetWindow call. You can ignore the pointer, although you may choose to check and verify its contents.

pAudioData

Pointer to audio data.

LICENSE AGREEMENT



REALNETWORKS, INC.

REDISTRIBUTION NOT PERMITTED

SDK License for the Visualization SDK

IMPORTANT -- READ CAREFULLY: This RealNetworks End User License Agreement (“License Agreement”) is a legal agreement between you (either an individual or an entity) and RealNetworks, Inc. and its suppliers and licensors (collectively “RN”) for RN's Visualization SDK (“SDK”) and the applicable documentation. By clicking on the “Accept” button, installing, copying, transmitting content, or otherwise using the SDK, you agree to be bound by the terms of this License Agreement. If you do not agree to the terms of this License Agreement, click on the “Cancel” button and/or do not install the SDK.

Any third party software that MAY BE provided with the SDK is included for use at your option. RN shall not be responsible for any losses OR damages which may occur resulting from the use of any THIRD PARTY SOFTWARE.

1. OWNERSHIP: This is a license agreement and NOT an agreement for sale. Title, ownership rights and intellectual property rights in and to the SDK (including any images, trademarks, animations, video, audio, music, and text incorporated into the SDK), accompanying printed materials, and any copies you are permitted to make herein are owned by RN or its suppliers and are protected by United States copyright law and international treaty provisions. Your rights to use the SDK are specified in this License Agreement, and RN retains all rights not expressly granted to you in this License Agreement. Nothing in this License Agreement constitutes a waiver of RN's rights under U.S. or international copyright law or any other federal or state law.

GRANT OF LICENSE:

(a) By RN. Subject to the provisions contained herein, RN hereby grants you a limited, non-exclusive, non-transferable license to: (i) use, pursuant to the documentation, **ONE** copy of the SDK on any single computer or on a second computer so long as the first and second computers are not used simultaneously;

and (ii) use the SDK to create visualizations that are compatible with RealJukebox and the SDK specifications (“Content”).

(b) By You. By uploading or otherwise transmitting any Content to RN, you automatically grant to RN a worldwide, royalty-free, perpetual, irrevocable, non-exclusive and fully sublicensable right and license to: (i) use, reproduce, transmit, retransmit, modify, adopt, publish, translate, create derivative works from, distribute, perform and display the Content in whole or part; (ii) incorporate such Content into other forms, media, or technology now known or later developed; (iii) use the Content and any ideas, concepts, know-how, techniques and suggestions contained in any Content for any purpose whatsoever, including, without limitation, developing, promoting and marketing the SDK, the Content, RN’s proprietary media delivery software applications (the “Software”), and any related services. You agree that by uploading or otherwise transmitting Content to RN you waive any rights of attribution or so-called “moral rights.”

3. RESTRICTIONS ON USE: You may not use the SDK to create Content: (i) with any material (artwork, logos, photographs, etc.) for which you have not obtained the necessary licenses and/or permissions; (ii) that contains defamatory or libelous statements or contains expressions of bigotry, prejudice, racism or profanity; (iii) that is primarily commercial in nature; (iv) that contains nudity, pornographic or obscene material of any kind; or (v) that promotes harm, injury or any act of cruelty to persons, property or animals. Further, you may not: (v) publish, distribute or upload Content that contains third-party material for which you have not obtained the necessary licenses and/or permissions; (vi) delete or modify any attributions, legal notices or other proprietary designations or labels on the SDK, or on any third-party material contained in the Content; (vii) use the SDK by itself or in conjunction with any other content creation specifications or products to make Content for any non-RN software application or service; (viii) copy the SDK, or any documentation accompanying the SDK except for back-up or archival purposes, provided any such copy must contain all of the original SDK's proprietary notices; or (ix) permit other individuals to use the SDK or to rent, lease, transfer, or otherwise transfer rights to the SDK or accompanying documentation. Any such forbidden use shall immediately terminate your license to the SDK.

4. ADDITIONAL RESTRICTIONS: You agree that the Content: (i) will meet RN’s technical and privacy requirements; (ii) may not be used to collect personally identifying information about end users; (iii) will not contain any computer code intentionally designed to disrupt, disable, harm or otherwise impede in any manner, including aesthetic disruptions or distortions, the

operation of the Software, firmware, computer systems or networks (sometimes referred to as “viruses” or “worms”); and (iv) must be developed to function with RN Products and not degrade the functionality of RN Products.

5. RN TRADEMARK LICENSE: RN hereby grants you a non-exclusive, limited license to use, and you agree that you shall always use, RN’s trademarks in accordance with RN’s Trademark and Logo Usage Policy at <http://www.real.com/company/guide/policy.html>, and for the sole purpose creating the Content. You agree that you shall not use any RN trademark in a way that may imply that you are an agency or branch of RN, or that RN endorses, is affiliated with, or sponsors you or your products. You also agree that you may not link directly to any media file or .ram file made available from the RN World Wide Website.

6. UPGRADES/SUPPORT: You shall not be entitled under this License Agreement to receive any updates, upgrades or corrections to the SDK, nor any support services.

7. DISCLAIMER OF WARRANTIES/LIMITATION OF LIABILITY: The option to upload or otherwise transmit Content to RN is made available to you as a courtesy. RN has no obligation to display or otherwise distribute any Content, to continue to display or distribute any Content after it has been posted, or to continue to make this service available. THE SDK, DOCUMENTATION, AND ANY SOFTWARE OR RELATED SERVICES MADE AVAILABLE BY RN IN CONNECTION WITH THE CREATION, DEVELOPMENT, UPLOADING, TRANSMITTING OR DISPLAYING ANY CONTENT ARE PROVIDED BY RN ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, RN FURTHER DISCLAIMS ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. THE ENTIRE RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE SOFTWARE AND DOCUMENTATION REMAINS WITH YOU. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL RN OR ITS SUPPLIERS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THIS AGREEMENT OR THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF RN HAS

BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES/JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

8. INDEMNIFICATION: This SDK is intended for use only with properly licensed media, content and content creation tools. It is your responsibility to ascertain whether any copyright, patent or other licenses are necessary and to obtain any licenses to such media and content. You agree to use only those materials for which you have the necessary patent, copyright and other permissions, licenses, and/or clearances. You agree to hold harmless, indemnify and defend RN, its officers, directors and employees, from and against any losses, damages, fines and expenses (including attorneys' fees and costs) arising out of or relating to any claims that: (i) you have breached or violated any term or condition of this Agreement; (ii) that the Content or uploading the Content infringes the intellectual property rights of any third party (including, without limitation, copyrights, trade secrets, patents, privacy rights, and trademark rights); (iii) that you have not obtained a license, right or permission to use material contained in the Content; (iv) that you do not have the rights necessary to grant RN the license set forth in Section 2 above; (v) the Content contains defamatory or libelous statements; expressions of bigotry, prejudice, racism or profanity; nudity, pornographic or obscene material of any kind; or that the Content promotes harm, injury or any act of cruelty to persons, property or animals; (vi) the Content contains harmful code; and (vii) the Content violates an end users privacy.

9. TERMINATION: This Agreement and your right to use this SDK automatically terminate if you fail to comply with any material provision of this Agreement. RN may terminate this License at any time by delivering notice to you and you may terminate this License at any time by destroying or erasing your copy of the SDK. Upon termination of this License Agreement, you agree to destroy or erase the SDK.

10. U.S. GOVERNMENT RESTRICTED RIGHTS: This SDK and documentation are provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer SDK--Restricted Rights at FAR 52.227-19 when applicable, or in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer SDK clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR supplement, as applicable. Manufacturer is RealNetworks, Inc./2601 Elliott Avenue/Suite 1000/Seattle, Washington, 98121.

You acknowledge that none of the SDK or underlying information or technology may be downloaded or otherwise exported or re-exported (i) into (or to a national or resident of) Cuba, Iraq, Libya, Sudan, North Korea, Yugoslavia, Iran, Syria or any other country to which the U.S. has embargoed goods; or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals or the U.S. Commerce Department's Table of Denial Orders. By using the SDK you are agreeing to the foregoing and are representing and warranting that you are not located in or under the control of a national or resident of any such country or on any such list.

11. MISCELLANEOUS: This License Agreement shall constitute the complete and exclusive agreement between us, notwithstanding any variance with any purchase order or other written instrument submitted by you, whether formally rejected by RN or not. The acceptance of any purchase order is you place is expressly made conditional on your consent to the terms set forth herein. The terms and conditions contained in this License Agreement may not be modified except in a writing duly signed by you and an authorized representative of RN. If any provision of this License Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable, and such decision shall not affect the enforceability of such provision under other circumstances, or of the remaining provisions hereof under all circumstances. This License Agreement shall be governed by the laws of the State of Washington, without regard to conflicts of law provisions, and you hereby consent to the exclusive jurisdiction of the state and federal courts sitting in the State of Washington. Any and all unresolved disputes relating in any way to, or arising out of, the Software, your use of the Software or this License Agreement shall be submitted to arbitration in the State of Washington; except that, to the extent that you have breached or have indicated your intention to breach this License Agreement in any manner which violates or may violate RN's intellectual property rights, or may cause continuing or irreparable harm to RN (including, but not limited to, any breach that may impact RN's intellectual property rights, or a breach by reverse engineering), RN may seek injunctive relief, or any other appropriate relief, in any court of competent jurisdiction. Any arbitration of a dispute under this Agreement shall be conducted under the rules then prevailing of the American Arbitration Association. The arbitrator's award shall be binding and may be entered as a judgment in any court of competent jurisdiction. This License Agreement will not be governed by the United Nations Convention of Contracts for the International Sale of Goods, the application of which is hereby expressly excluded.

Copyright © 1999-2000 RealNetworks, Inc. and/or its suppliers. 2601 Elliott Avenue, Suite 1000, Seattle, Washington 98121 U.S.A. All rights reserved. RealNetworks, RealAudio, RealVideo, RealMedia, RealPlayer, and RealJukebox are trademarks or registered trademarks of RealNetworks, Inc.