NeroVision API

v3.1.0.22

The *NeroVision API* will only work with fully installed *Nero* and *NeroVision* versions!

v1.10, Copyright 2004-2005 Nero AG

Nero AG, Im Stoeckmaedle 18, 76307 Karlsbad, Germany

1 Contents

1 Contents ... 2
2 License Agreement ... 7
3 Introduction ... 8
3.1 Overview ... 8
3.2 Requirements ... 8
3.3 The NeroSDK Forum ... 8
4 Quick Start ... 9
4.1 Running the Precompiled NeroVisionAPIExample Application ... 9
4.2 Compiling the NeroVisionAPIExample ... 11
4.3 Accessing *NeroVision API* in Your Applications ... 11
4.4 Points of Consideration ... 13
5 The NeroVisionAPIExample Application ... 14
5.1 AudioContentStream ... 14
5.2 AudioProvider ... 14
5.3 BaseContentStream ... 14
5.4 BaseObj ... 14
5.5 CAboutDlg ... 14
5.6 CEstimateDoneDlg ... 15
5.7 CEstimatingDlg ... 15
5.8 CNVAPIExampleApp ... 15
5.9 CNVAPIExampleDlg ... 15
5.10 CNVAPIExamplePage ... 15
5.11 ContentResolver ... 15
5.12 CPage ... 15
5.13 CPageData ... 16
5.14 CPageExport ... 16
5.15 CPageFirst ... 16
5.16 CPagePreview ... 16
5.17 CPageSecond ... 16
5.18 CPageThird ... 17
5.19 CRecorderCombobox ... 17
5.20 CSheet ... 17
5.21 CSpeedComboBox ... 18
5.22 CSplashThread ... 18
5.23 StreamSample ... 18
5.24 VideoContentStream ... 18
5.25 VideoProvider ... 18
6 An Example Project ... 19
6.1 The Project Components ... 19
6.2 The XML File ... 20
6.3 The Slide Show ... 21
6.4 The Video Title ... 21
6.5 The Menu ... 22
7 Interfaces and Types ... 23
7.1 IError Interface ... 23
7.1.1 IError::ErrCode Enum ... 23

7.1.2 IError::ErrCode Property ... 23
7.1.3 IError::ErrText Property ... 23
7.1.4 IError::XMLID Property ... 23
7.2 INeroBurnContext Interface ... 24
7.2.1 INeroBurnContext::NERO_CD_FORMAT Property ... 24
7.2.2 INeroBurnContext::NERO_WRITE_CD Property ... 24
7.3 IProgressCallback Interface ... 24
7.3.1 IProgressCallback::ProgressAction Enum ... 24
7.3.2 IProgressCallback::OnProgress Event ... 24
7.3.3 IProgressCallback::ShouldCancel Event ... 24
7.4 IProject Interface ... 25
7.4.1 IProject::SetXMLString Method ... 25
7.4.2 IProject::SetXMLFile Method ... 25
7.4.3 IProject::EstimateDiskSize Method ... 25
7.4.4 IProject::EstimateCreateNeroBurnContextTime Method ... 25
7.4.5 IProject::CreateNeroBurnContext Method ... 25
7.4.6 IProject::TempDirectory Property ... 25
7.4.7 IProject::LastError Property ... 25
7.5 IProject2 ... 26
7.5.1 IProject2::ContentResolver Property ... 26
7.6 IProject3 ... 26
7.6.1 IProject3::SupportedXMLSchema Property ... 26
7.7 IMovieExporter ... 26
7.7.1 IMovieExporter::SetSource Method ... 26
7.7.2 IMovieExporter::EstimateFileSize Method ... 26
7.7.3 IMovieExporter::ExportMovie Method ... 26
7.7.4 IMovieExporter::LastError Method ... 27
7.8 IEffectEnumerator ... 27
7.8.1 IEffectEnumerator::EffectCategory Enum ... 27
7.8.2 IEffectEnumerator::NumEffects Property ... 27
7.8.3 IEffectEnumerator::GetEffectBaseInfo Method ... 27
7.8.4 IEffectEnumerator::LastError Property ... 27
7.9 IMediaInfo Interface ... 28
7.9.1 IMediaInfo::FileType Enum ... 28
7.9.2 IMediaInfo::FileType Property ... 28
7.9.3 IMediaInfo::Duration Property ... 28
7.9.4 IMediaInfo::HasAudio Property ... 28
7.9.5 IMediaInfo::HasVideo Property ... 28
7.9.6 IMediaInfo::Width Property ... 29
7.9.7 IMediaInfo::Height Property ... 29
7.9.8 IMediaInfo::AspectXByY Property ... 29
7.9.9 IMediaInfo::FrameRate Property ... 29
7.9.10 IMediaInfo::GetThumbnail Method ... 29
7.9.11 IMediaInfo::GetImage Method ... 29
7.10 IFileInfo Interface ... 29
7.10.1 IFileInfo::FilePath Property ... 29
7.10.2 IFileInfo::NumSegments Property ... 29
7.10.3 IFileInfo::GetSegmentMediaInfo Method ... 30

7.11 IMediaAnalyzer Interface 30
7.11.1 IMediaAnalyzer::AddFile Method 30
7.11.2 IMediaAnalyzer::AnalyzeFiles Method 30
7.11.3 IMediaAnalyzer::LastError Property 30
7.12 Preview component 30
7.12.1 IPreviewCallback 30
7.12.1.1 IPreviewCallback::NotifyPosChanged Event 31
7.12.1.2 IPreviewCallback::NotifyPlaybackEnd Event 31
7.12.2 IPreview 31
7.12.2.1 IPreview::SimulationKey Enum 31
7.12.2.2 IPreview::SimulationState Enum 32
7.12.2.3 IPreview::SetPreviewContext Method 32
7.12.2.4 IPreview::SetPreviewWindow Method 32
7.12.2.5 IPreview::BackgroundBitmap Property 32
7.12.2.6 IPreview::BackgroundColor Property 32
7.12.2.7 IPreview::NotifyOwnerMessage Method 32
7.12.2.8 IPreview::StartPreview Method 33
7.12.2.9 IPreview::StopPreview Method 33
7.12.2.10 IPreview::JumpToID Method 33
7.12.2.11 IPreview::JumpToChapter Method 33
7.12.2.12 IPreview::GetCurrentChapter Method 33
7.12.2.13 IPreview::Duration Property 33
7.12.2.14 IPreview::Volume (in) Property 33
7.12.2.15 IPreview::Volume (out) Property 33
7.12.2.16 IPreview::Position (in) Property 34
7.12.2.17 IPreview::Position (out) Property 34
7.12.2.18 IPreview::State Property Method 34
7.12.2.19 IPreview::HandleKeyInput Method 34
7.12.2.20 IPreview::HandleNumericInput Method 34
7.12.2.21 IPreview::IsKeySupported Method 34
7.12.2.22 IPreview::Callback Property 34
7.12.2.23 IPreview::LastError Property 34
7.13 Content Provider 35
7.13.1 Types 35
7.13.1.1 ContentType 35
7.13.1.2 AudioFormat 35
7.13.1.3 VideoFormat 35
7.13.1.4 FrameStructure 35
7.13.1.5 VideoInfo 35
7.13.1.6 AudioInfo 36
7.13.2 IContentProvider 36
7.13.2.1 IContentProvider::ProvidedContentType Property 36
7.13.3 IContentResolver 36
7.13.3.1 IContentResolver::ResolveContent Method 36
7.13.4 IAVStreamSample 36
7.13.4.1 IAVStreamSample::GetTime Method 36
7.13.4.2 IAVStreamSample::GetData Method 36
7.13.5 IAVStream 37

7.13.5.1 IAVStream::SetPosition Method 37
7.13.5.2 IAVStream::GetSample Method 37
7.13.6 IAVStreamProvider 37
7.13.6.1 IAVStreamProvider::GetStream Method 37
7.13.6.2 IAVStreamProvider::GetDuration Method 37
7.13.7 IAudioStreamProvider 37
7.13.7.1 IAudioStreamProvider::GetInfo Method 37
7.13.8 IVideoStreamProvider 37
7.13.8.1 IVideoStreamProvider::GetInfo 37
7.14 Project Object 38
7.15 MediaAnalyzer Object 38
7.16 MovieExporter Object 38
7.17 EffectEnumerator Object 38
7.18 Preview Object 38
8 The XML Schema Definition 39
8.1 element nerovision-project 39
8.2 element nerovision-project/vcd 40
8.3 element nerovision-project/svcd 41
8.4 element nerovision-project/dvd 42
8.5 complexType DiscProjectBaseType 43
8.6 element DiscProjectBaseType/label 43
8.7 element DiscProjectBaseType/content 44
8.8 element videotitle 44
8.9 element videotitle/video 45
8.10 element videotitle/audio 45
8.11 element videotitle/chapters 46
8.12 element videotitle/chapters/chapter-mark 46
8.13 complexType AVSettingsBaseType 47
8.14 element DiscProjectBaseType/default-av-settings 47
8.15 element DiscProjectBaseType/menu 47
8.16 element DiscProjectBaseType/menu/default-template 49
8.17 element DiscProjectBaseType/menu/default-appearance 49
8.18 element DiscProjectBaseType/menu/main-menu 50
8.19 element DiscProjectBaseType/menu/main-menu/template 50
8.20 element DiscProjectBaseType/menu/main-menu/appearance 51
8.21 element DiscProjectBaseType/menu/main-menu/thumbnail 51
8.22 element chapter-menu 52
8.23 element chapter-menu/template 53
8.24 element chapter-menu/appearance 53
8.25 element chapter-menu/thumbnail 54
8.26 element slideshow 55
8.27 element slideshow/image 56
8.28 element slideshow/image/header 56
8.29 element slideshow/image/footer 57
8.30 element slideshow/audio 57
8.31 element transition 58
8.32 complexType MenuAppearanceBaseType 58
8.33 element MenuAppearanceBaseType/background 59

8.34 element MenuAppearanceBaseType/background/solid-color 59
8.35 element MenuAppearanceBaseType/background/image 60
8.36 element MenuAppearanceBaseType/background/video 60
8.37 element MenuAppearanceBaseType/background/audio 61
8.38 element MenuAppearanceBaseType/foreground 61
8.39 element MenuAppearanceBaseType/foreground/header 62
8.40 element MenuAppearanceBaseType/foreground/footer 62
8.41 element MenuAppearanceBaseType/foreground/buttons 63
8.42 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons 64
8.43 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/auto-layout 64
8.44 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout 65
8.45 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/playitem-button 65
8.46 element MenuAppearanceBaseType/foreground/buttons/previous-button 66
8.47 element MenuAppearanceBaseType/foreground/buttons/next-button 66
8.48 element MenuAppearanceBaseType/foreground/buttons/return-button 67
8.49 complexType MenuTemplateBaseType 67
8.50 complexType ThumnailBaseType 68
8.51 element ThumnailBaseType/image 68
8.52 element ThumnailBaseType/video 68
8.53 simpleType AVTime 69
8.54 simpleType FileLocation 69
8.55 simpleType Percentage 69
8.56 simpleType RGBColor 69
8.57 simpleType VideoEncoding 70
8.58 simpleType EffectType 70
8.59 attributeGroup AVFileAttrGrp0 70
8.60 attributeGroup AVFileAttrGrp1 71
8.61 attributeGroup AVFileAttrGrp2 71
8.62 attributeGroup FontAttrGrp0 72
8.63 attributeGroup FontAttrGrp1 72
8.64 attributeGroup RelativeCoordAttrGrp0 72
8.65 attributeGroup RelativeCoordAttrGrp1 72
9 FAQ 73
10 Known Limitations 75

2 License Agreement

IMPORTANT: PLEASE READ THE SOFTWARE LICENSE AGREEMENT ("LICENSE") CAREFULLY BEFORE USING THE SOFTWARE.

USING THE SOFTWARE INDICATES YOUR ACKNOWLEDGMENT THAT YOU HAVE READ THE LICENSE AND AGREE TO ITS TERMS.

The license agreement is contained in a text file, “NeroSDK_License.txt”, to be found in the root folder of the installation package.

3 Introduction

3.1 Overview

This paper, the documentation of *NeroVision API*, contains some practical guidelines on how to use the *NeroVision* API (Application Programming Interface) with the help of *NeroSDK* (Nero Software Development Kit).

The SDK is available for OEMs (Original Equipment Manufacturers) and registered users of *Nero*.

NeroVision API enables you to use a number *NeroVision 2* features in your own applications:

- Creation of slide shows
- Creation of menus
- Creation of video tracks
- Menu simulation
- Encoding preview
- Export of videos and slideshows
- Enumeration of effects

Supported formats:

- Video CD
- Super Video CD
- DVD

3.2 Requirements

This documentation assumes that *Nero* 6.3.0.0 or later and *NeroVision* 3.1.0.22 or later are already installed on your computer.

3.3 The NeroSDK Forum

We provide a forum for all users of the *NeroSDK* to get in dialog with each other at http://www.nero.com/link.php?topic_id=17. We will also monitor the messages from time to time and try to help where possible.

4 Quick Start

The main effort in working with *NeroVision API* is the creation of an XML file which contains information about your VCD, SVCD or DVD project, while the use of the functional API itself is pretty straight-forward. The XML file is required to adhere to an XML Schema as documented later in this paper.

Once you created your project in XML format it can be passed to *NeroVision API* for processing.

The API estimates processing time and provides feedback on current phase and progress.

4.1 Running the Precompiled NeroVisionAPIExample Application

We advise you to use the example application provided with the API to familiarize yourself with the use.

To run the example, it is required that MS-XML 3.0 or better is installed.

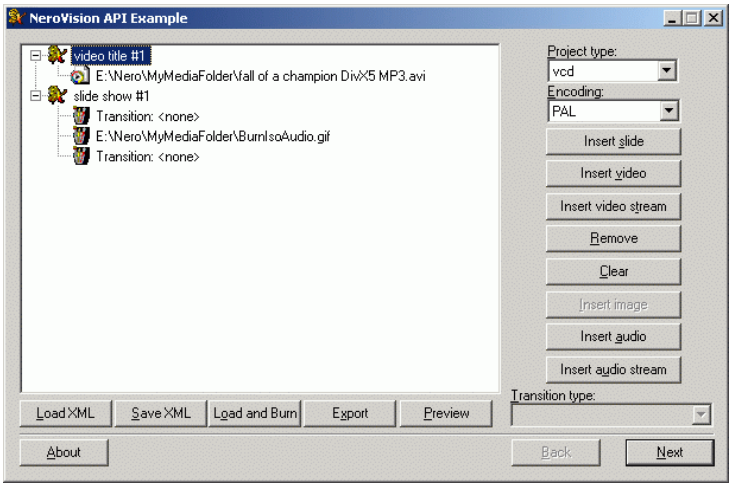
The example is a Wizard-style dialog application that leads the user through project creation and API use with a number of tabs.

On the first page, a *NeroVision API* project can be loaded, saved and configured.

The user can choose between VCD, SVCD or DVD projects and add slide shows or videos. In addition, the transition type between individual content elements can be set.

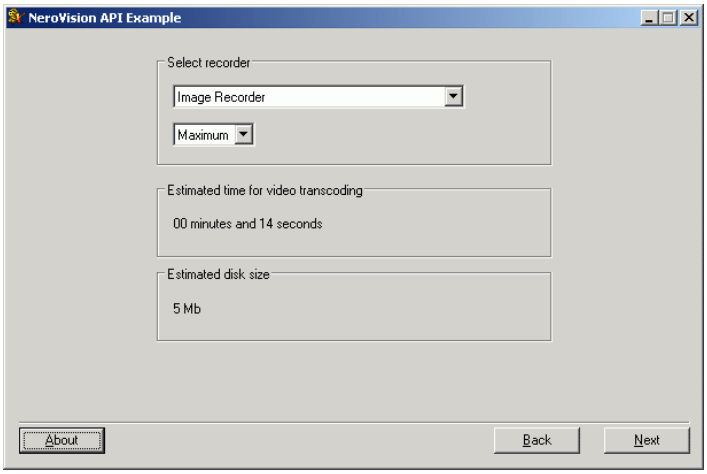
Audio tracks can be assigned to slide shows or videos.

The user can burn, export and preview the project.

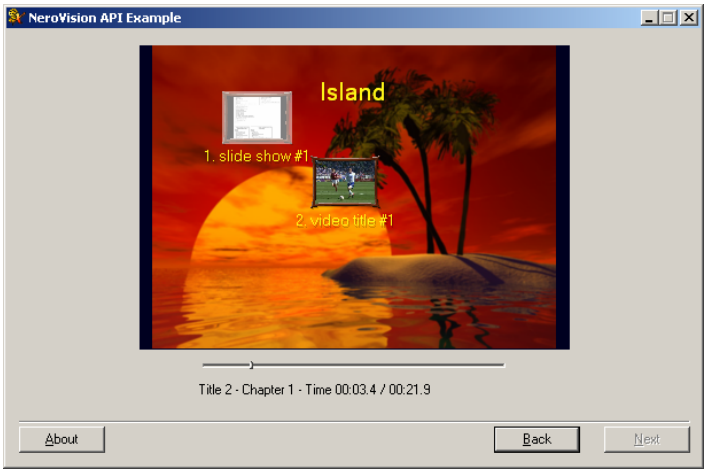


The second page offers a recorder and speed selection dialog.

The second page is the one where *NeroVision API* is actually used to provide an estimate for transcoding time and disk size.

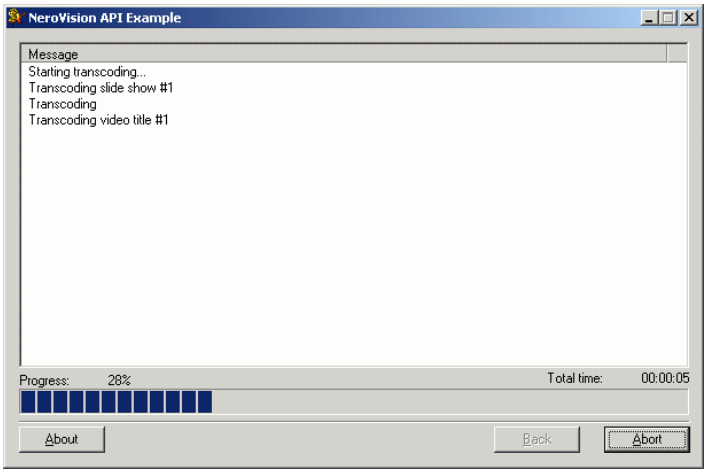


The preview page offers the opportunity to experience the generated menu structure, view images, play video titles and basically perform all operations that will be allowed on the target media.



The third page provides the user with feedback on the current transcoding or burn progress.

The *NeroAPI* is used to burn the project on the desired recordable media.



Finally, the example will report the successful completion of the operation.

4.2 Compiling the NeroVisionAPIExample

The NeroVisionAPIExample project might not compile as intended if due to localization issues or personal preference your actual configuration differs from the default path for English versions of your Operating System.

Therefore it might be necessary to modify the source code and adjust the path to the *NeroVision API* DLL before you compile the project.

Mainly, the following path in the StdAfx.h header file should be altered to match whatever location is correct in your actual environment:

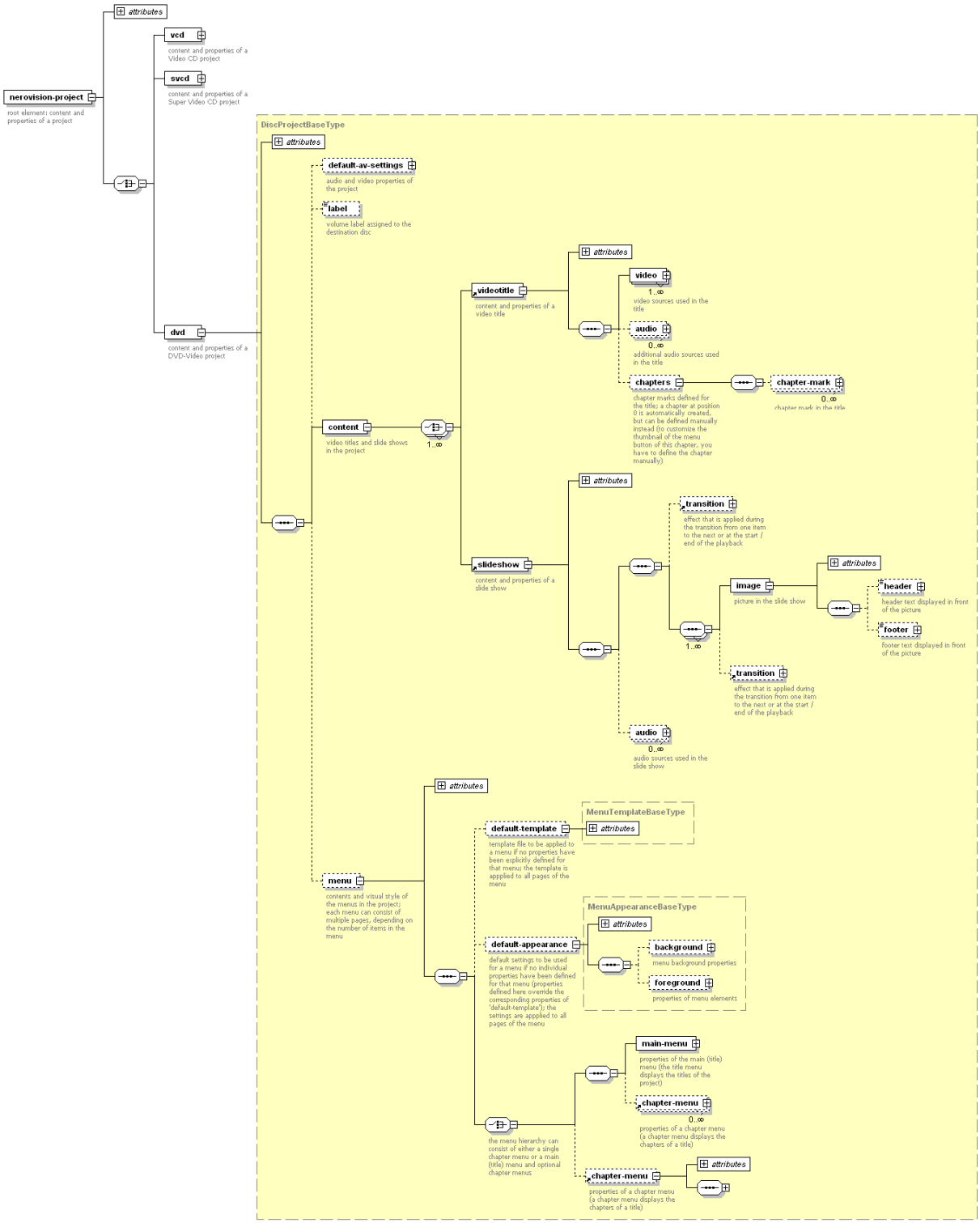
```
#import "C:\Program Files\Ahead\NeroVision\NeroVisionAPI.dll"
```

4.3 Accessing *NeroVision API* in Your Applications

In general the following steps are required:

- Make the files of the *NeroAPI*-include-directory accessible from your program.
- Link your project with the NeroAPIGlue library.
- #import from the NeroVisionAPI.dll
- Use the *NeroAPI* functions in proper order.
- Implement the *NeroAPI* and *NeroVision API* callback functions.
- Create an XML project that complies with the schema definition and pass it to *NeroVision API*
- Ask *NeroVision API* to process the project.
- Handle the feedback from *NeroVision API*, e.g. visualizing progress for the user.
- Use the *NeroAPI* to burn your project.

Below you can see the DVD branch of the schema partially unfolded. This shall give you an impression of the complexity you will deal with. The description of the structure as outlined by the schema is part of this documentation together with an annotated example XML project file.



4.4 Points of Consideration

Make sure that the application will find the required DLLs, by installing *Nero* and *NeroVision*.

Provide a working infrastructure for XML processing, e.g. through MS-XML.

5 The NeroVisionAPIExample Application

The NeroVisionAPIExample illustrates a number of important aspects of working with *NeroVision API*:

- Creating a project in XML format or loading it from hard disk
- Passing the project to *NeroVision API* and getting estimates
- Obtaining device information through the *NeroAPI*
- Processing the project and handing it over to the *NeroAPI*
- Collecting video and audio data from streams
- Exporting MPEG video
- Previewing the entire *NeroVision* project

The example is a Wizard-style dialog application that leads the user through XML project creation and *NeroVision API* use with a number of tabs.

The NeroVisionAPIExample project uses MS-XML to create an XML project file that can be processed by *NeroVision API*.

5.1 AudioContentStream

This is a simple audio stream implementation. The generated audio stream will contain the stream name in Morse code.

5.2 AudioProvider

This class is the provider for an audio stream.

5.3 BaseContentStream

This class represents the base implementation of IAVStream; the classes VideoContentStream and AudioContentStream are derived from this class.

5.4 BaseObj

This is the base class for COM objects.

5.5 CAboutDlg

This is a standard about box.

5.6 CEstimateDoneDlg

This dialog box implementation contains controls that show the results of transcoding estimations for disk space and time.

5.7 CEstimatingDlg

This dialog displays the current state of the transcoding estimation progress for disk usage and transcoding time.

5.8 CNVAPIExampleApp

This is a CWinApp derived class.

In InitInstance *NeroAPI* is initialized through calls to NeroAPIGlueConnect and NeroInit. An object of the CNVAPIExampleDlg class is instantiated.

The ExitInstance function contains the code to clean up *NeroAPI*, in particular through calls to NeroDone and NeroAPIGlueDone. Apart from that, CoUninitialize is called.

This class also contains NeroIdleCallback and NeroUserDialog callback implementations that forward the callbacks to the currently active window and return its response.

5.9 CNVAPIExampleDlg

This class is an implementation of the main property-sheet-like dialog.

5.10 CNVAPIExamplePage

This template class adds a function for obtaining the CPageData pointer to each class that derives from it.

5.11 ContentResolver

The main method of this class will resolve a content provider ID to the respective content type. For this test application any ID is valid.

If the ID begins with "A:", the method returns an audio provider, all other IDs resolve to a video provider while empty IDs cause an error.

5.12 CPage

This class contains the declaration of the common page abstract class.

It is not instantiated directly; rather than that it is implemented in the derived classes CPageFirst, CPageSecond, CPageThird and CPagePreview.

5.13 CPageData

This class maintains common data to be communicated among pages:

- The raw XML data string m_sXML
- The NERO_SCSI_DEVICE_INFO pointer m_pDeviceInfo
- The burn speed value m_dwBurnSpeed
- A pointer to the project interface in m_pProject
- The selected tree item m_selectedItemIndex

The constructor creates an instance of *NeroVision API*'s Project object.

5.14 CPageExport

This dialog class serves the purpose of exporting a video in MPEG format through the IMovieExporter interface.

5.15 CPageFirst

This class represents the first page in the dialog where the project is set up. CPageFirst contains all the XML parsing code.

The dialog allows the user to add videos and slideshows. Once a slideshow element has been created, individual images can be attached to it.

The tree items on the first level can be of the CONTENTTYPE type, where CONTENTTYPE is VIDEOTITLE or SLIDESHOW.

The tree items on the second level of a SLIDESHOW can be of the TRANSITION type. INVALID means an image path. NONE means no transition.

The two integers that hold the sequence numbers for newly inserted video and slideshow items are used for automatic unique naming of new items.

The BuildXML method is used to build an XML tree from the project data.

5.16 CPagePreview

This dialogs controls the preview capability of the example application.

During Preview, the current time and position will be displayed.

5.17 CPageSecond

When the second dialog is opened, *NeroVision API* will be asked to estimate the processing time for the current project.

The central function of this class is OnChangeState.

Depending on direction from the page has been entered (coming from page one or page three), different actions will be taken.

If the page is activated for the first time, the recorder combobox is filled and the speeds combobox is updated accordingly.

If the page is activated in the forward direction, the dialog will prepare the GUI for the display of transcoding time and disk size estimations. For a potentially failed estimation it will load and set an error string.

Then, the XML string, as constructed on page one, will be injected into *NeroVision API*'s Project object. If XML setting succeeded, the application will ask for disk size and transcoding duration estimates.

Obtaining the transcoding duration estimation might take a while. Therefore, we are passing an IProgressCallback interface pointer which we are implementing. This will give us an opportunity to move the progress bar and indicate how long the process will take.

5.18 CPageThird

On the third page the process is started and the user is kept updated on the current progress. The currently processed item is displayed, along with the remaining time.

The central function of this class is OnStartProcess. The CreateNeroBurnContext method will not return until transcoding finishes but we have a progress callback to monitor the process. If an error occurred, a corresponding message will be displayed.

After a successful method call, finally NeroAPI’s NeroBurn function will be called to perform the actual burning.

5.19 CRecorderCombobox

This is a class for a combobox that lists all available recorders.

The FillCombo function will determine the available drives through *NeroAPI,* determine their characteristics and fill the combo box list accordingly.

When the control is destroyed, the device infos are freed by a call to NeroFreeMem.

The GetCurrentRecorder method will retrieve the currently selected recorder. The actual NERO_SCSI_DEVICE_INFO pointer is stored in item's data pointer.

5.20 CSheet

This is the common sheet abstract class. It is implemented in CNVAPIExampleDlg.

5.21 CSpeedComboBox

This is a combobox class that holds all available speeds for a particular recorder.

The RefreshSpeeds method is used to refresh the speeds of the combobox, using the recorder that was passed as parameter. It utilizes *NeroAPI*'s NeroGetAvailableSpeeds function to obtain the available speeds for the device.

5.22 CSplashThread

This class contains the implementation of a modal splash window that is displayed during the application's initialization phase.

5.23 StreamSample

Implementation of IAVStreamSample. It is used in BaseContentStream to create individual samples.

5.24 VideoContentStream

This class implements a video stream; the generated video frames contain the stream name and the frame number.

5.25 VideoProvider

The provided test stream has the following properties:

- RGB32
- 25 fps
- aspect ratio 4:3
- progressive

The video size defaults to 480x360 but can be changed for testing by specifying the desired size at the beginning of the stream name

6 An Example Project

The following is an example of an XML project file for a Super Video CD project with a slideshow containing two images with audio track, one video title and a menu.

The details of the individual elements are described in the Schema Definition section of this document.

6.1 The Project Components

The project consists of a nerovision-project element which provides the frame. This project is of type “svcd” and contains a label.

The slide show and video title are embedded in the <content> element. The menu has its own XML element tag <menu>.

XML version information

The project

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<nerovision-project version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="NeroVisionAPI.xsd">

  <svcd encoding="PAL">

    <label>This is a label!</label>
    <content>

      Slide show      Video title

    </content>

    Menu

  </svcd>

</nerovision-project>
```

Super Video CD

6.2 The XML File

Here is the fully expanded view of the project file. The individual segments will be explained on the following pages.

Project Frame

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<nerovision-project version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="NeroVisionAPI.xsd">
    <svcd encoding="PAL">

        <label>This is a label!</label>
        <content>
```

Slide show with audio

```
            <slideshow name="Rome burns" id="s1">
                <transition type="WipeUp"/>
                <image src="D:\MyMediaFolder\TrulyBeautifulPicture.bmp" duration="5">
                    <header>Image header</header>
                </image>
                <transition type="WipeDown"/>
                <image src="D:\MyMediaFolder\AnotherGreatPicture.bmp" duration="5">
                    <header font-color="#000000" font-face="Arial" font-size="24">Header</header>
                    <footer font-color="#000000">Footer</footer>
                </image>
                <transition type="WipeRight"/>
                <audio src="D:\MyMediaFolder\LivingOnVideo.wav"/>
            </slideshow>
```

Video title

```
            <videotitle name="Nero fiddles" id="v1">
                <video src="D:\MyMediaFolder\GreatActionMovie.avi"/>
                <chapters>
                    <chapter-mark pos="120" id="cm1" name="Interesting chapter"/>
                    <chapter-mark pos="180" id="cm2" name="Interesting chapter"/>
                </chapters>
            </videotitle>
        </content>
```

Menu

```
        <menu default-thumbnail="FirstNonEmptyFrame">
            <default-template id="templ1" name="Island"/>
            <main-menu/>
            <chapter-menu title-id="v1"/>
        </menu>
    </svcd>
</nerovision-project>
```

6.3 The Slide Show

The slide show consists of two images with an audio track.

The first image will appear with a "wipe up" transition and contains a header text. This text will display in the upper part of the image in standard system font.

After 5 seconds the second image will appear with a "wipe down" transition. It contains a header with color code for black, font face Arial and a size of 24.

The footer will be black as well. The other font characteristics will be taken from the standard system font.

The second image will disappear with a "wipe right" transition after 5 seconds.

```
<slideshow name="Rome burns" id="s1">

  <transition type="WipeUp"/>
  <image src="D:\MyMediaFolder\TrulyBeautifulPicture.bmp" duration="5">
     <header>Image header</header>
  </image>

  <transition type="WipeDown"/>
  <image src="D:\MyMediaFolder\AnotherGreatPicture.bmp" duration="5">
     <header font-color="#000000" font-face="Arial" font-size="24">Header</header>
     <footer font-color="#000000">Footer</footer>
  </image>

  <transition type="WipeRight"/>

  <audio src="D:\MyMediaFolder\LivingOnVideo.wav"/>

</slideshow>
```

Transition and first image with header

Transition and second image with header and footer

Audio track for slideshow

6.4 The Video Title

The Super Video CD project contains one video title which consists of one video file. Two chapter marks are given. The first one is 120 seconds from the start, the second one 180 seconds.

```
<videotitle name="Nero fiddles" id="v1">

  <video src="D:\MyMediaFolder\GreatActionMovie.avi"/>

  <chapters>
     <chapter-mark pos="120" id="cm1" name="Interesting chapter"/>
     <chapter-mark pos="180" id="cm2" name="Interesting chapter"/>
  </chapters>

</videotitle>
```

Chapter mark definition

6.5 The Menu

The menu contains links to the individual chapters of the video title. By default the chapters will be represented by a thumbnail image of the first non empty frame.

```
<menu default-thumbnail="FirstNonEmptyFrame">
    <default-template id="templ1" name="Island"/>
    <main-menu/>
    <chapter-menu title-id="v1"/>
</menu>
```

Reference to template

Reference to video title

7 Interfaces and Types

This chapter describes the interface to the *NeroVision API* DLL.

7.1 IError Interface

NeroVision API error codes and error description.

7.1.1 IError::ErrCode Enum

The error code is an enumeration type with the following constants:

Description of enumerators	
NoError	No error.
Canceled	Canceled by user.
InternalError	Internal error.
APIUsageError	API usage error.
InstallationError	An error with the installation.
XMLSyntaxError	Incorrect XML syntax.
FileIOError	Error during file I/O.
MenuTemplateError	Menu template error.
NeroAPIError	An error while working with the NeroAPI.
TranscodingFailed	Transcoding could not be finished successfully.
DiscOverflow	Disc overflow.
ContentResolverError	Error in the IContentResolver interface.
ContentProviderError	Error in the IContentProvider interface.
DiscStructureError	The disc structure contains an error.
PreviewError	Error during preview.
EffectError	Error during effect creation.

7.1.2 IError::ErrCode Property

Gets the error code.

```
HRESULT ErrCode([out, retval] ErrorCode* errCode);
```

7.1.3 IError::ErrText Property

Gets the error text.

```
HRESULT ErrText([out, retval] BSTR* errStr);
```

7.1.4 IError::XMLID Property

Gets the ID of the XML tag involved in the error.

```
HRESULT XMLID([out, retval] BSTR* xmlID);
```

7.2 INeroBurnContext Interface

The result of CreateNeroBurnContext.

7.2.1 INeroBurnContext::NERO_CD_FORMAT Property

Gets the resulting NERO_CD_FORMAT (2nd parameter of NeroBurn()).

```
HRESULT NERO_CD_FORMAT([out, retval] int* pFormat);
```

7.2.2 INeroBurnContext::NERO_WRITE_CD Property

Gets pointer to the resulting NERO_WRITE_CD structure (3rd parameter of NeroBurn()).

```
HRESULT NERO_WRITE_CD([out, retval] void** pNWCD);
```

7.3 IProgressCallback Interface

Interface for handling progress information.

7.3.1 IProgressCallback::ProgressAction Enum

Action reported by ProgressCallback.

Description of enumerators	
Estimating	Estimating time for processing.
Preparing	Preparing data for transcoding.
Transcoding	Transcoding.
Generating	Generating.
Analyzing	Analyzing.

7.3.2 IProgressCallback::OnProgress Event

Called regularly to report progress of the current sub-task and overall progress of the invoked function.

```
HRESULT OnProgress([in] ProgressAction action, [in] BSTR itemname, [in] double
currentRemain, [in] double currentTotal, [in] float currentFraction, [in]
double totalRemain, [in] double totalTotal, [in] float totalFraction);
```

7.3.3 IProgressCallback::ShouldCancel Event

Called regularly to check whether the called function should be canceled.

```
HRESULT ShouldCancel([out, retval] boolean* pbCancel);
```

7.4 IProject Interface

A *NeroVision API* project describes a DVD/(S)VCD.

7.4.1 IProject::SetXMLString Method

Sets the project content from raw XML data.

```
HRESULT SetXMLString([in] const char* data, [out, retval] boolean* pbSuccess);
```

7.4.2 IProject::SetXMLFile Method

Sets the project content from a XML file.

```
HRESULT SetXMLFile([in] BSTR filepath, [out, retval] boolean* pbSuccess);
```

7.4.3 IProject::EstimateDiskSize Method

Estimates the size of the resulting disk.

```
HRESULT EstimateDiskSize([out] hyper* size, [out, retval] boolean* pbSuccess);
```

7.4.4 IProject::EstimateCreateNeroBurnContextTime Method

Estimates the time needed by CreateNeroBurnContext().

```
HRESULT EstimateCreateNeroBurnContextTime([in] IProgressCallback* pCallback,
[out] double* seconds, [out, retval] boolean* pbSuccess);
```

7.4.5 IProject::CreateNeroBurnContext Method

This method creates a NeroBurnContext object which can be burned with Nero. If a ProgressCallback is passed, it will receive progress information.

```
HRESULT CreateNeroBurnContext([in] HMODULE hNeroAPIDLL, [in] IProgressCallback*
pCallback, [out] INeroBurnContext** ppburnContext, [out, retval] boolean*
pbSuccess);
```

7.4.6 IProject::TempDirectory Property

This method sets the directory used for storing temporary files. Defaults to the system temp. directory.

```
HRESULT TempDirectory([in] BSTR path);
```

7.4.7 IProject::LastError Property

Gets the error object of the last called function.

```
HRESULT LastError([out, retval] IError** pError);
```

7.5 IProject2

Extended interface of IProject.

```
interface IProject2 : IProject
```

7.5.1 IProject2::ContentResolver Property

Sets a ContentResolver for resolving references to virtual content in the XML.

```
HRESULT ContentResolver([in] IContentResolver* resolver);
```

7.6 IProject3

Extended interface of IProject.

```
interface IProject3 : IProject2
```

7.6.1 IProject3::SupportedXMLSchema Property

Returns the supported XML schema.

```
HRESULT SupportedXMLSchema([out, retval] BSTR* schema);
```

7.7 IMovieExporter

Interface for exporting a movie title to a file.

```
interface IMovieExporter : IUnknown
```

7.7.1 IMovieExporter::SetSource Method

Sets the movie to export, which can be either a whole movie project or a videotitle/slideshow of a (S)VCD/DVD project.

```
HRESULT SetSource([in] IProject* prj, [in] BSTR xmlID, [out, retval] boolean*
pbSuccess);
```

7.7.2 IMovieExporter::EstimateFileSize Method

Estimates the size of the resulting file.

```
HRESULT EstimateFileSize([out] hyper* size, [out, retval] boolean* pbSuccess);
```

7.7.3 IMovieExporter::ExportMovie Method

Exports the movie to file. If a ProgressCallback is passed, it will receive progress information.

```
HRESULT ExportMovie([in] BSTR filepath, [in] IProgressCallback* pCallback,
[out, retval] boolean* pbSuccess);
```

7.7.4 IMovieExporter::LastError Method

Gets the error object of the last called function.

```
HRESULT LastError([out, retval] IError** pError);
```

7.8 IEffectEnumerator

Interface for enumerating the installed effects.

```
interface IEffectEnumerator : IUnknown
```

7.8.1 IEffectEnumerator::EffectCategory Enum

The categories of an effect.

```
typedef enum
{
  EffectCategory_Video,
  EffectCategory_Transition,
  EffectCategory_Text
} EffectCategory;
```

7.8.2 IEffectEnumerator::NumEffects Property

Retrieves the number of effects installed.

```
HRESULT NumEffects([out, retval] int* numEffects);
```

7.8.3 IEffectEnumerator::GetEffectBaseInfo Method

Obtains information about an effect. 'typeid' is the unique identifier ('type' attribute in XML).

```
HRESULT GetEffectBaseInfo([in] int effectIndex, [out] BSTR* typeID, [out] BSTR*
displayName, [out] EffectCategory* category, [out, retval] boolean* pbSuccess);
```

7.8.4 IEffectEnumerator::LastError Property

Retrieves the error object of the last called function.

```
HRESULT LastError([out, retval] IError** pError);
```

7.9 IMediaInfo Interface

Attributes and methods for audio and video of a single media.

7.9.1 IMediaInfo::FileType Enum

MediaInfo file type.

```
typedef enum
{
  Invalid = 0,
  Other,
  MPEG1,
  MPEG2,
  DV_DV,
  DV_AVI1,
  DV_AVI2
} FileType;
```

Description of enumerators	
Invalid	Invalid file type.
Other	Other file type.
MPEG1	MPEG1 file.
MPEG2	MPEG2 file.
DV_DV	DV_DV file.
DV_AVI1	DV_AVI1 file.
DV_AVI2	DV_AVI2 file.

7.9.2 IMediaInfo::FileType Property

Media file type.

```
HRESULT FileType([out, retval] FileType* ft);
```

7.9.3 IMediaInfo::Duration Property

Duration in seconds.

```
HRESULT Duration([out, retval] double* sec);
```

7.9.4 IMediaInfo::HasAudio Property

Determines whether the Media has audio.

```
HRESULT HasAudio([out, retval] boolean* has);
```

7.9.5 IMediaInfo::HasVideo Property

Determines whether the Media has video.

```
HRESULT HasVideo([out, retval] boolean* has);
```

7.9.6 IMediaInfo::Width Property

Media width.

```
HRESULT Width([out, retval] int* w);
```

7.9.7 IMediaInfo::Height Property

Media height.

```
HRESULT Height([out, retval] int* h);
```

7.9.8 IMediaInfo::AspectXByY Property

Media TV system.

```
HRESULT AspectXByY([out, retval] double* aspect);
```

7.9.9 IMediaInfo::FrameRate Property

Frame time in seconds.

```
HRESULT FrameRate([out, retval] double* rate);
```

7.9.10 IMediaInfo::GetThumbnail Method

Retrieves the first (non-empty) thumbnail of the media.

```
HRESULT GetThumbnail([in] boolean ensureNotEmpty, [out, retval] long* hDIB);
```

7.9.11 IMediaInfo::GetImage Method

Retrieves a frame of the media at a given position.

```
HRESULT GetImage([in] double posSec, [out, retval] long* hDIB);
```

7.10 IFileInfo Interface

Attributes and methods for a physical file, which may contain several segments.

7.10.1 IFileInfo::FilePath Property

The associated file's path and name.

```
HRESULT FilePath([out, retval] BSTR* filepath);
```

7.10.2 IFileInfo::NumSegments Property

Retrieves the number of segments in the file.

```
HRESULT NumSegments([out, retval] int* numSeg);
```

7.10.3 IFileInfo::GetSegmentMediaInfo Method

Gets the MediaInfo for one segment of the file.

```
HRESULT GetSegmentMediaInfo([in] int segIndex, [out, retval] IMediaInfo**
pMediaInfo);
```

7.11 IMediaAnalyzer Interface

Interface for analyzing one or more media files.

7.11.1 IMediaAnalyzer::AddFile Method

Add a file to be analyzed. The resulting IFileInfo is filled after calling AnalyzeFiles().

```
HRESULT AddFile([in] BSTR filepath, [out, retval] IFileInfo** fileInfo);
```

7.11.2 IMediaAnalyzer::AnalyzeFiles Method

Analyzes all files previously added by AddFile().

```
HRESULT AnalyzeFiles([in] IProgressCallback* pCallback, [out, retval] boolean*
pbSuccess);
```

7.11.3 IMediaAnalyzer::LastError Property

Gets the error object of the last called function.

```
HRESULT LastError([out, retval] IError** pError);
```

7.12 Preview component

The Preview component of *NeroVisionAPI* can be used to

- preview a single video or title, or
- simulate the behaviour of a complete NeroVision project including the menu.

The preview component uses its own window to display the content of videos and projects. The window can be integrated into a client application as a child window.

Basically, the preview component handles user input and events by itself. However, the client application can use a callback mechanism to get notified on certain events. The client application is responsible for notifying the preview component of certain user actions, like a position change of the parent window.

The preview component supports various navigation commands. The availability of each navigation command depends on the current state of the preview or simulation.

7.12.1 IPreviewCallback

This interface is passed to the preview component in order to receive notifications during the playback or simulation.

7.12.1.1 IPreviewCallback::NotifyPosChanged Event

Called when the playback position has changed.

```
HRESULT NotifyPosChanged([in] double pos, [in] int title, [in] int chapter);
```

7.12.1.2 IPreviewCallback::NotifyPlaybackEnd Event

Called when the playback has finished.

```
HRESULT NotifyPlaybackEnd();
```

7.12.2 IPreview

This interface is used to preview the contents of a project or title.

7.12.2.1 IPreview::SimulationKey Enum

Input keys supported by the simulation component.

Description	
SimulationKey_Play	Starts the playback.
SimulationKey_Pause	Halts the playback at the current position.
SimulationKey_Up	Moves the selection to the upper next item.
SimulationKey_Down	Moves the selection to the lower next item.
SimulationKey_Left	Moves the selection to the adjacent item in left direction.
SimulationKey_Right	Moves the selection to the adjacent item in right direction.
SimulationKey_Enter	Activates the current selection.
SimulationKey_RootMenu	Jumps back into the main menu.
SimulationKey_TitleMenu	Jumps back into the current title menu.
SimulationKey_Prev	Jumps to the beginning of the previous title or chapter.
SimulationKey_Next	Jumps to the beginning of the next title or chapter.
SimulationKey_Return	Stops the simulation.
SimulationKey_Numeric	Activates the button that has the index of the entered numeric value.

7.12.2.2 IPreview::SimulationState Enum

Defines the current playback state of the simulation component.

Description	
SimulationState_Stopped	The simulation is stopped.
SimulationState_Paused	The simulation is paused.
SimulationState_Running	The simulation is running.

7.12.2.3 IPreview::SetPreviewContext Method

Assigns the project and item to be previewed.

```
HRESULT SetPreviewContext([in] IProject* project, [in] BSTR xmlID, [out,
retval] boolean* pbSuccess);
```

7.12.2.4 IPreview::SetPreviewWindow Method

Specifies the placement of the preview window.

```
HRESULT SetPreviewWindow([in] HANDLE_PTR hParentWnd, [in] int left, [in] int
top, [in] int right, [in] int bottom, [out, retval] boolean* pbSuccess);
```

7.12.2.5 IPreview::BackgroundBitmap Property

Specifies a picture that is to be displayed when the playback is stopped. The caller is responsible for freeing the bitmap handle.

```
HRESULT BackgroundBitmap([in] HANDLE_PTR hDib);
```

7.12.2.6 IPreview::BackgroundColor Property

Specifies a uniform color that is to be displayed when the playback is stopped. The color is not displayed if a background bitmap has also been specified. However, it is always used to fill the areas that are not covered by the video window or background bitmap. The default color is black.

```
HRESULT BackgroundColor([in] COLORREF color);
```

7.12.2.7 IPreview::NotifyOwnerMessage Method

Must be called to relay certain Windows messages to the preview component.

```
HRESULT NotifyOwnerMessage([in] unsigned int uMsg, [in] long wParam, [in] long
lParam);
```

7.12.2.8 IPreview::StartPreview Method

Starts the playback or simulation.

```
HRESULT StartPreview([out, retval] boolean* pbSuccess);
```

7.12.2.9 IPreview::StopPreview Method

Stops the playback or simulation.

```
HRESULT StopPreview();
```

7.12.2.10 IPreview::JumpToID Method

Sets the playback position to the beginning of the specified item, represented by its XML ID.

```
HRESULT JumpToID([in] BSTR xmlID, [out, retval] boolean* pbSuccess);
```

7.12.2.11 IPreview::JumpToChapter Method

Sets the playback position to the specified chapter.

```
HRESULT JumpToChapter([in] int title, [in] int chapter, [out, retval] boolean*
pbSuccess);
```

7.12.2.12 IPreview::GetCurrentChapter Method

Returns the indices of the current chapter and title.

```
HRESULT GetCurrentChapter([out] int* title, [out] int* chapter);
```

7.12.2.13 IPreview::Duration Property

Returns the total duration of the title that is currently previewed.

```
HRESULT Duration([out, retval] double* duration);
```

7.12.2.14 IPreview::Volume (in) Property

Sets the audio volume.

```
HRESULT Volume([in] double volume);
```

7.12.2.15 IPreview::Volume (out) Property

Returns the current audio volume.

```
HRESULT Volume([out, retval] double* volume);
```

7.12.2.16 IPreview::Position (in) Property

Sets the playback position within the current title.

```
HRESULT Position([in] double pos);
```

7.12.2.17 IPreview::Position (out) Property

Returns the current playback position within the current title.

```
HRESULT Position([out, retval] double* pos);
```

7.12.2.18 IPreview::State Property Method

Returns the current state of the simulation.

```
HRESULT State([out, retval] SimulationState* state);
```

7.12.2.19 IPreview::HandleKeyInput Method

Notifies the simulation component of keyboard input.

```
HRESULT HandleKeyInput([in] SimulationKey key);
```

7.12.2.20 IPreview::HandleNumericInput Method

Notifies the simulation component of keyboard input of a numeric key.

```
HRESULT HandleNumericInput([in] unsigned char num);
```

7.12.2.21 IPreview::IsKeySupported Method

Used to determine whether the specified key is supported by the simulation component in the current configuration.

```
HRESULT IsKeySupported([in] SimulationKey key, [out, retval] boolean*
supported);
```

7.12.2.22 IPreview::Callback Property

Assigns a callback interface to the preview component.

```
HRESULT Callback([in] IPreviewCallback* notify);
```

7.12.2.23 IPreview::LastError Property

Returns the error object created by the last called function.

```
HRESULT LastError([out, retval] IError** pError);
```

7.13 Content Provider

The content provider interface provides methods to base content creation on stream input.

7.13.1 Types

7.13.1.1 ContentType

The type of virtual content.

```
typedef enum
{
  AudioStream = 0,
  VideoStream
} ContentType;
```

7.13.1.2 AudioFormat

The supported audio formats for virtual content.

```
typedef enum
{
  AudioFormat_PCM = 0
} AudioFormat;
```

7.13.1.3 VideoFormat

The supported video formats for virtual content.

```
typedef enum
{
  VideoFormat_YV12 = 0,
  VideoFormat_RGB32,
  VideoFormat_RGB24
} VideoFormat;
```

7.13.1.4 FrameStructure

The frame structure of the video content.

```
typedef enum
{
  FrameStructure_Progressive = 0,
  FrameStructure_InterlacedTopFirst,
  FrameStructure_InterlacedBottomFirst
} FrameStructure;
```

7.13.1.5 VideoInfo

Description of virtual video content.

```
typedef struct
{
  VideoFormat viFormat;
  FrameStructure viStructure;
```

```
  int viWidth;
  int viHeight;
  hyper viTimePerFrame;
  int viAspectX;
  int viAspectY;
} VideoInfo;
```

7.13.1.6 AudioInfo

Description of virtual audio content.

```
typedef struct
{
  AudioFormat aiFormat;
  int aiSamplesPerSec;
  short aiBitsPerSample;
  short aiNumChannels;
} AudioInfo;
```

7.13.2 IContentProvider

This is the base interface for feeding virtual content into *NeroVision API*.

```
interface IContentProvider : IUnknown
```

7.13.2.1 IContentProvider::ProvidedContentType Property

Gets the type of content provided.

```
HRESULT ProvidedContentType([out, retval] ContentType* pCT);
```

7.13.3 IContentResolver

Interface for creating a content provider for a given content-id

```
interface IContentResolver : IUnknown
```

7.13.3.1 IContentResolver::ResolveContent Method

```
HRESULT ResolveContent([in] BSTR id, [out, retval] IContentProvider** ppCP);
```

7.13.4 IAVStreamSample

Description of virtual video content.

```
interface IAVStreamSample: IUnknown
```

7.13.4.1 IAVStreamSample::GetTime Method

```
HRESULT GetTime([out] hyper* start, [out] hyper* stop);
```

7.13.4.2 IAVStreamSample::GetData Method

```
HRESULT GetData([out] DWORD* size, [out] void const** dataPtr);
```

7.13.5 IAVStream

Interface of an audio or video stream.

```
interface IAVStream : IUnknown
```

7.13.5.1 IAVStream::SetPosition Method

```
HRESULT SetPosition([in] const hyper* pos);
```

7.13.5.2 IAVStream::GetSample Method

```
HRESULT GetSample([out, retval] IAVStreamSample** sample);
```

7.13.6 IAVStreamProvider

Base interface for feeding audio or video content into *NeroVision API*.

```
interface IAVStreamProvider : IContentProvider
```

7.13.6.1 IAVStreamProvider::GetStream Method

```
HRESULT GetStream([out, retval] IAVStream** stream);
```

7.13.6.2 IAVStreamProvider::GetDuration Method

```
HRESULT GetDuration([out, retval] hyper* dur);
```

7.13.7 IAudioStreamProvider

Interface for feeding audio content into *NeroVision API*.

```
interface IAudioStreamProvider : IAVStreamProvider
```

7.13.7.1 IAudioStreamProvider::GetInfo Method

```
HRESULT GetInfo([out] AudioInfo* info);
```

7.13.8 IVideoStreamProvider

Interface for feeding video content into *NeroVision API*.

```
interface IVideoStreamProvider : IAVStreamProvider
```

7.13.8.1 IVideoStreamProvider::GetInfo

```
HRESULT GetInfo([out] VideoInfo* info);
```

7.14 Project Object

An object that represents a *NeroVision API* project.

```
coclass Project
{
   [default] interface IProject;
};
```

7.15 MediaAnalyzer Object

A Media analyzer object.

```
coclass MediaAnalyzer
{
  [default] interface IMediaAnalyzer;
};
```

7.16 MovieExporter Object

Movie exporter object.

```
coclass MovieExporter
{
  [default] interface IMovieExporter;
};
```

7.17 EffectEnumerator Object

Effect enumerator object.

```
coclass EffectEnumerator
{
  [default] interface IEffectEnumerator;
};
```

7.18 Preview Object

```
coclass Preview
{
  [default] interface IPreview;
};
```

8 The XML Schema Definition

The W3C defines an XML Schema like this:

"XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structure, content and semantics of XML documents."

An XML Schema represents the rules for elements and attributes that can appear in a document, including their data types and default and fixed values. It also defines which elements are child elements, and regulates their order and number.

Schema **NeroVisionAPI.xsd**

Elements	Complex types	Simple types	Attr. groups
chapter-menu	AVSettingsBaseType	AVTime	AVFileAttrGrp0
nerovision-project	DiscProjectBaseType	EffectType	AVFileAttrGrp1
slideshow	MenuAppearanceBaseType	FileLocation	AVFileAttrGrp2
transition	MenuTemplateBaseType	Percentage	FontAttrGrp0
videotitle	ThumnailBaseType	RGBColor	FontAttrGrp1
		VideoEncoding	RelativeCoordAttrGrp0

In the following we will list the elements and attributes as they appear in the schema, also noting their uses.

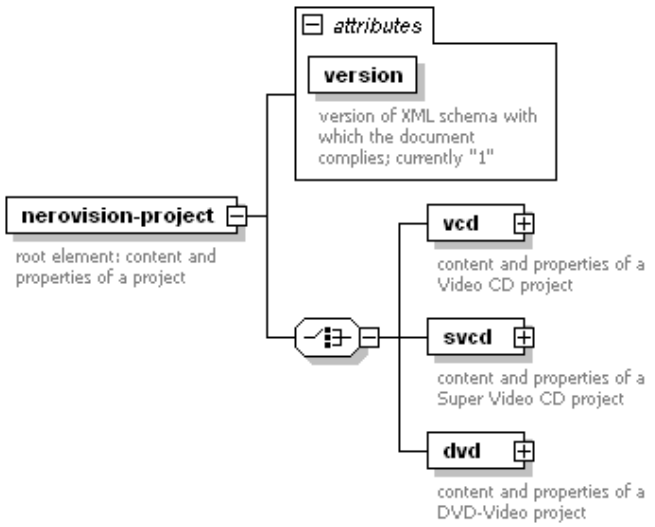
8.1 element nerovision-project

This is the top level element of any project. A project can either be a Video CD, Super Video CD or DVD.

Though the target media have different formats the structure of the project as accepted by *NeroVision API* is identical.

You will see that the elements representing them all are of the ProjectBaseType type.

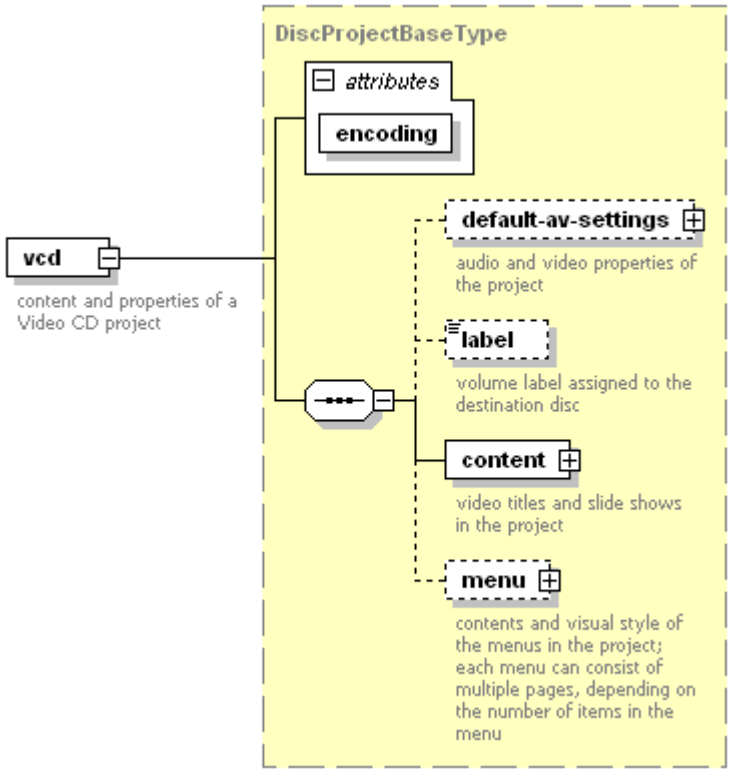
diagram



properties	content complex				
children	**vcd svcd dvd**				
attributes	Name	Type	Use	Default	Fixed
	version	**derived by: xs:int**	required		1

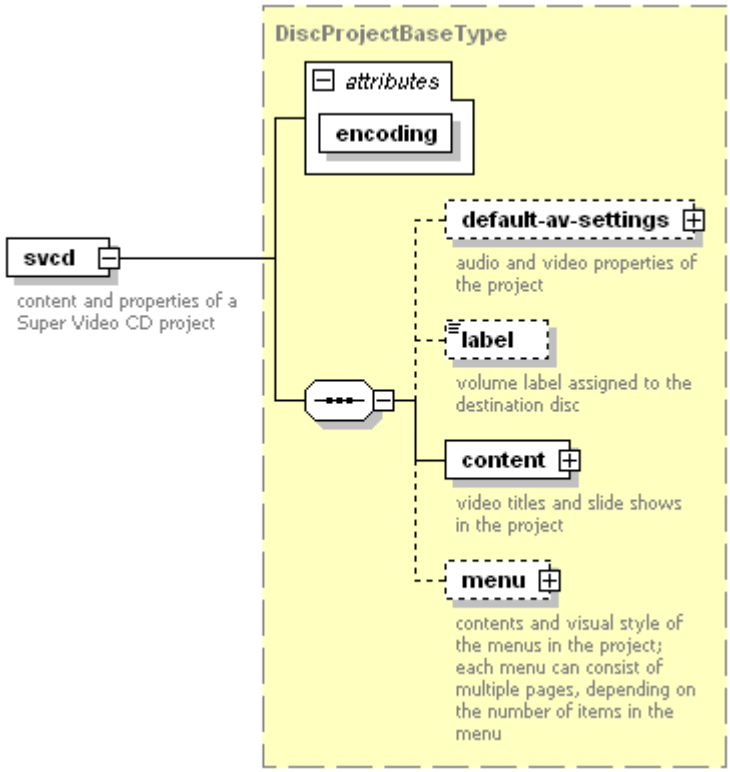
8.2 element nerovision-project/vcd

A Video CD project.

diagram					
type	**DiscProjectBaseType**				
properties	isRef 0 content complex				
children	**default-av-settings label content menu**				
attributes	Name	Type	Use	Default	Fixed
	encoding	**VideoEncoding**	required		

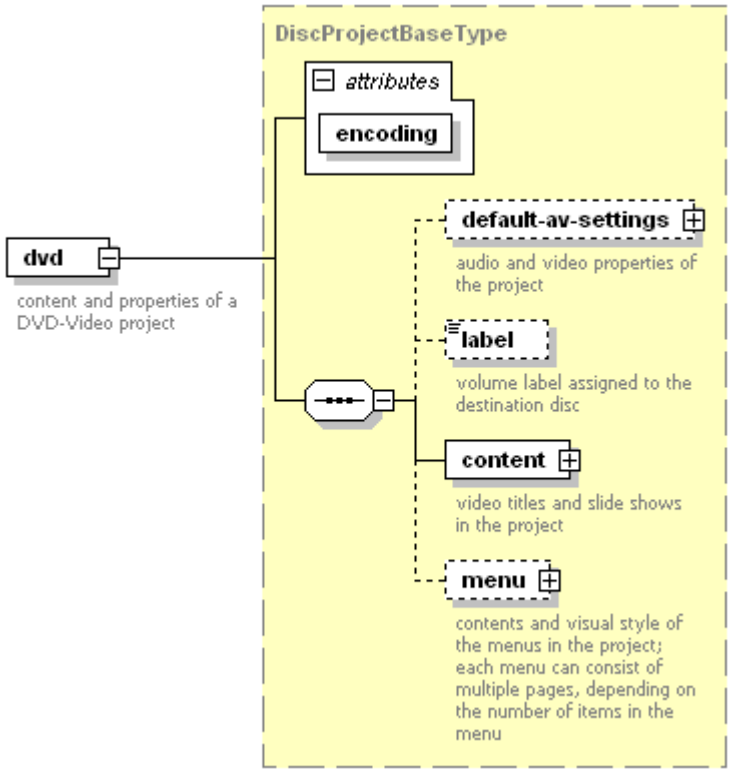
8.3 element nerovision-project/svcd

A Super Video CD project.

diagram	
type	**DiscProjectBaseType**
properties	isRef 0 content complex
children	**default-av-settings label content menu**
attributes	Name: encoding; Type: **VideoEncoding**; Use: required; Default: ; Fixed:

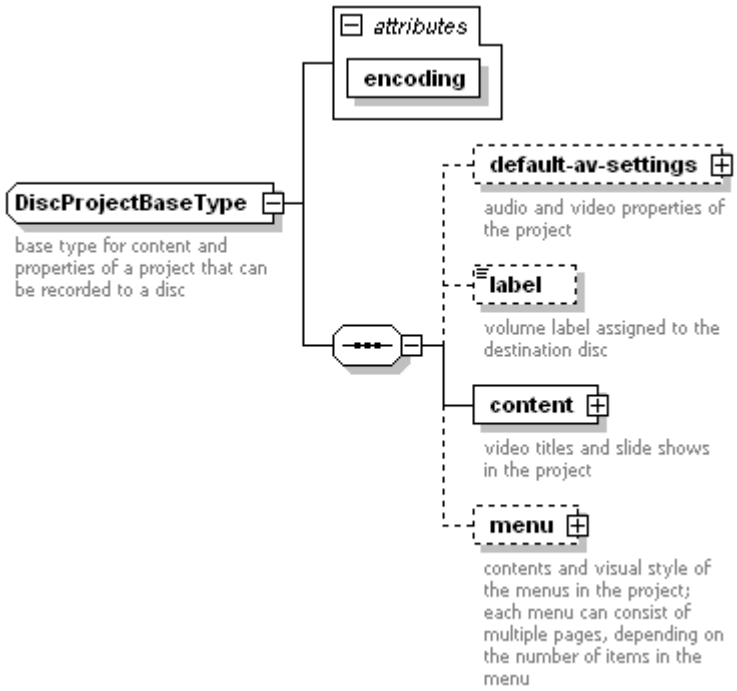
8.4 element nerovision-project/dvd

A DVD project.

diagram	
type	**DiscProjectBaseType**
properties	isRef 0 content complex
children	**default-av-settings label content menu**
attributes	Name: encoding; Type: **VideoEncoding**; Use: required; Default: ; Fixed:

8.5 complexType DiscProjectBaseType

This type represents a generic project. An actual project might be a Video CD, Super Video CD or DVD. All those are based on ProjectBaseType.

diagram	
children	**default-av-settings label content menu**
used by	elements **nerovision-project/dvd nerovision-project/svcd nerovision-project/vcd**
attributes	Name: encoding; Type: **VideoEncoding**; Use: required; Default: ; Fixed:

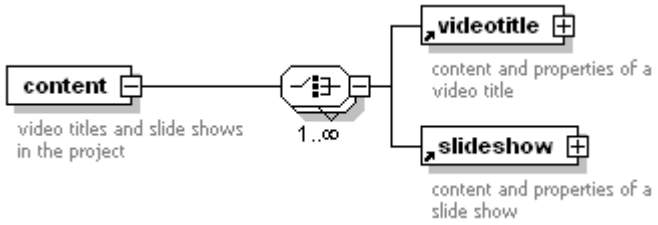
8.6 element DiscProjectBaseType/label

An optional label that can be attached to a project.

diagram	
type	restriction of **xs:string**
properties	isRef 0 minOcc 0 maxOcc 1 content simple
facets	maxLength 32

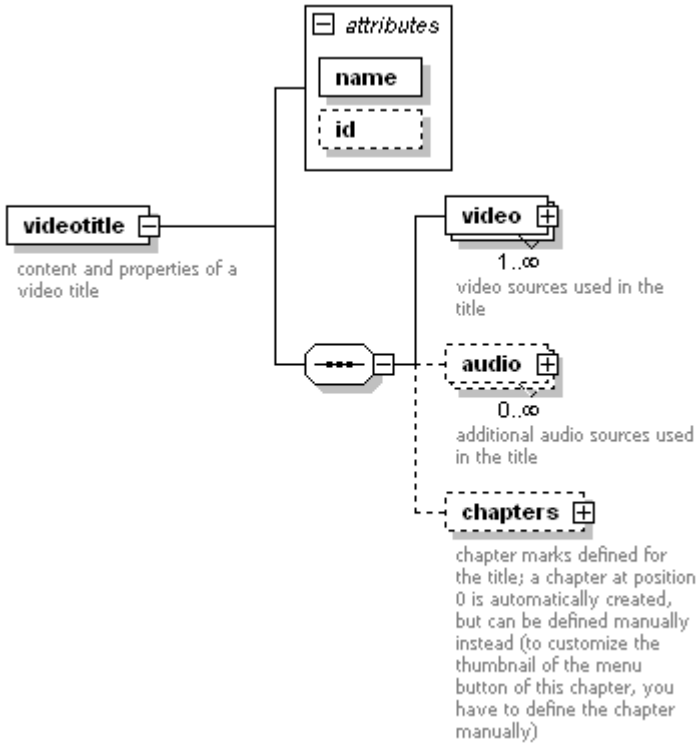
8.7 element DiscProjectBaseType/content

The project content can be a sequence of video titles and slideshows.

diagram	
properties	isRef 0 content complex
children	**videotitle slideshow**

8.8 element videotitle

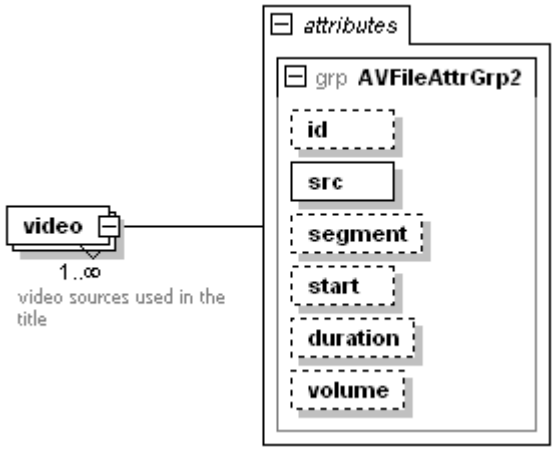
A video title represents a collection of individual video files. This collection can be addressed by addressed by the ID for the video title. The video title can be segmented in chapters.

diagram					
properties	content complex				
children	**video audio chapters**				
used by	element **DiscProjectBaseType/content**				
attributes	Name	Type	Use	Default	Fixed
	name	**xs:string**	required		
	id	**xs:ID**	optional		

8.9 element videotitle/video

A single video file. The ID field can be used to provide a unique identifier.

diagram



properties

isRef	0
minOcc	1
maxOcc	unbounded
content	complex

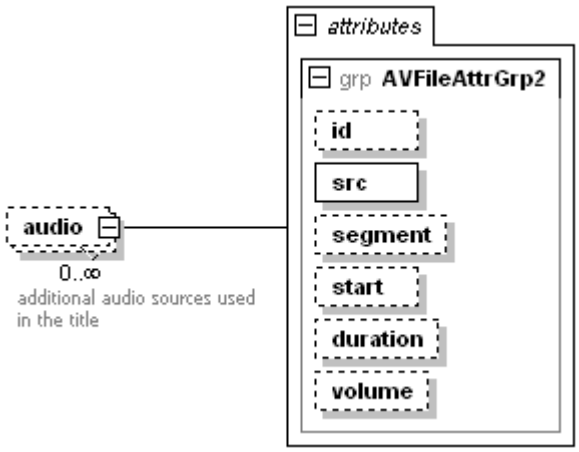
attributes

Name	Type	Use	Default	Fixed
id	**xs:ID**	optional		
src	**FileLocation**	required		
segment	**derived by: xs:int**	optional		
start	**AVTime**	optional	0	
duration	**AVTime**	optional		
volume	**Percentage**	optional	100%	

8.10 element videotitle/audio

Video titles can be combined with audio tracks, e.g. if the video media file does not already contain audio information.

diagram



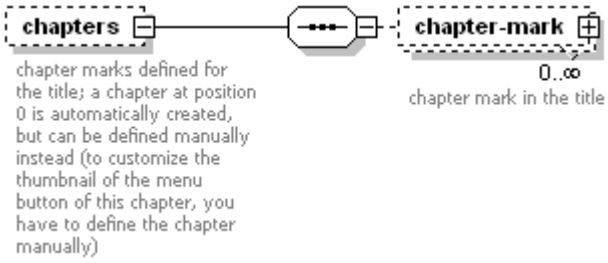
properties

isRef	0
minOcc	0
maxOcc	unbounded
content	complex

attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	required		
	segment	**derived by: xs:int**	optional		
	start	**AVTime**	optional	0	
	duration	**AVTime**	optional		
	volume	**Percentage**	optional	100%	

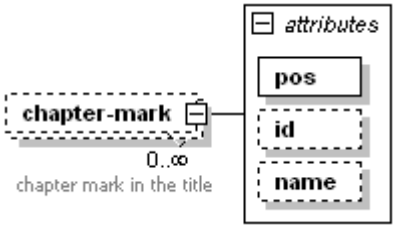
8.11 element videotitle/chapters

A sequence of chapter marks to structure a video title.

diagram	
properties	isRef 0 minOcc 0 maxOcc 1 content complex
children	**chapter-mark**

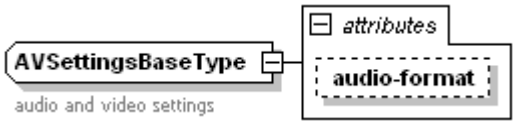
8.12 element videotitle/chapters/chapter-mark

A chapter-mark references a significant position in a video title.

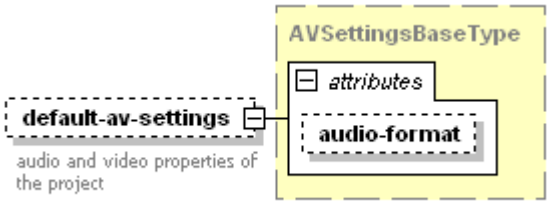
diagram	
properties	isRef 0 minOcc 0 maxOcc unbounded content complex

attributes	Name	Type	Use	Default	Fixed
	pos	**AVTime**	required		
	id	**xs:ID**	optional		
	name	**xs:string**	optional		

8.13 complexType AVSettingsBaseType

diagram	
used by	element **DiscProjectBaseType/default-av-settings**
attributes	Name: audio-format; Type: **derived by: xs:string**; Use: optional; Default: Automatic; Fixed:

8.14 element DiscProjectBaseType/default-av-settings

diagram	
type	**AVSettingsBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
attributes	Name: audio-format; Type: **derived by: xs:string**; Use: optional; Default: Automatic; Fixed:

8.15 element DiscProjectBaseType/menu

A menu consists either of

- a hierarchical main-menu with a number of chapter-menus as sub menus or a chapter-menu or
- a single chapter-menu

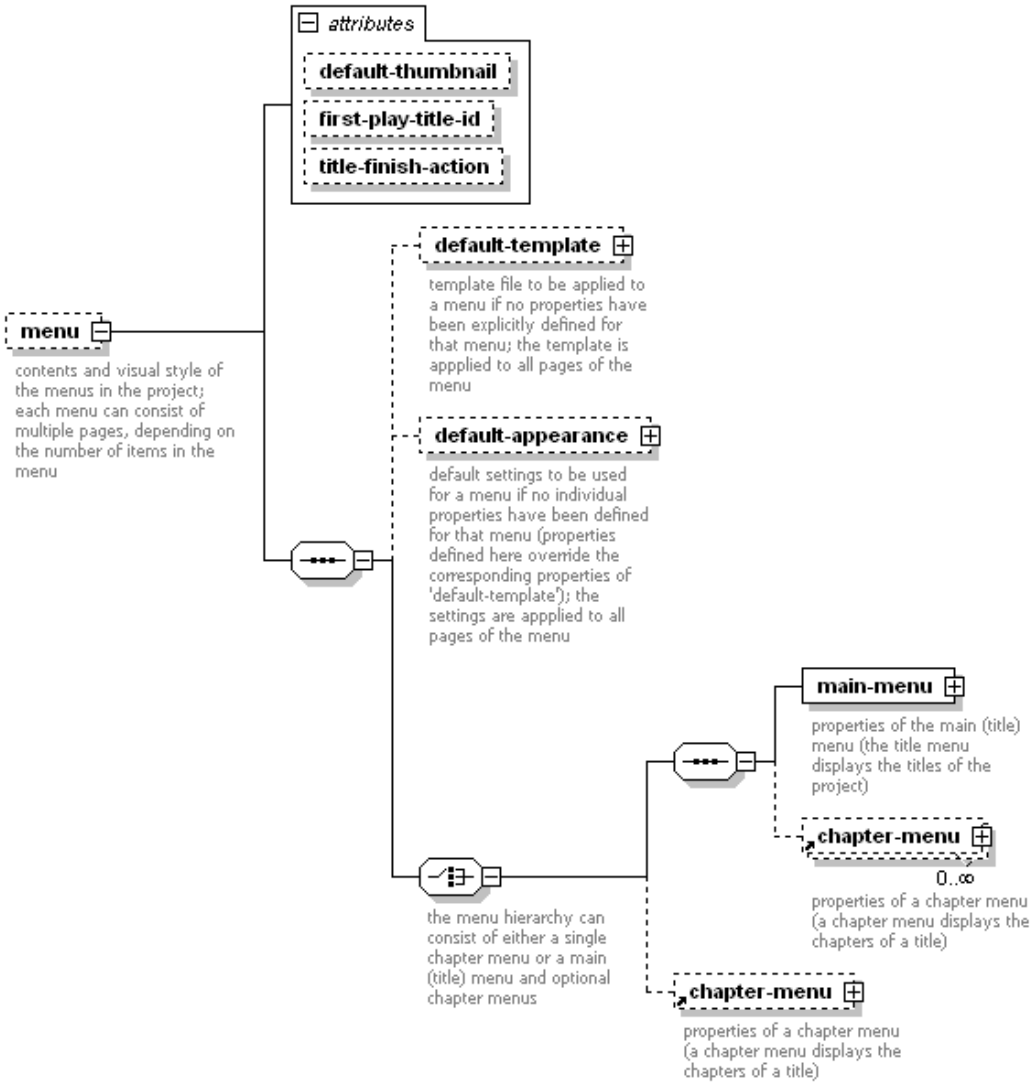
Slide shows do not have chapter-menus.

The default-thumbnail attribute should be set to FirstFrame or FirstNonEmptyFrame.

The title-finish-action attribute determines what happens after a title was played successfully. Allowed values are “PlayNextTitle”, “JumpMainMenu” and “LoopTitle”.

“LoopTitle” is only available if the project has a menu. “LoopTitle” is not permissible in DVD-VR and DVD-VFR projects.

diagram



properties

isRef	0
minOcc	0
maxOcc	1
content	complex

children

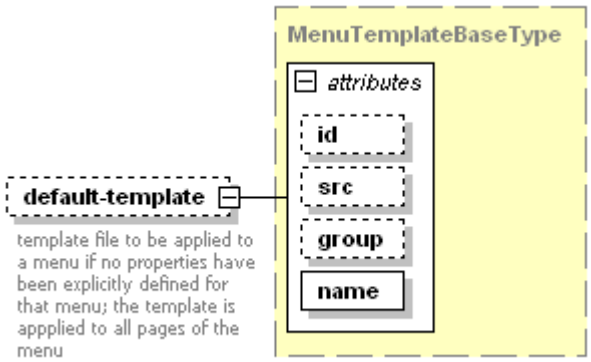
default-template default-appearance main-menu chapter-menu

attributes

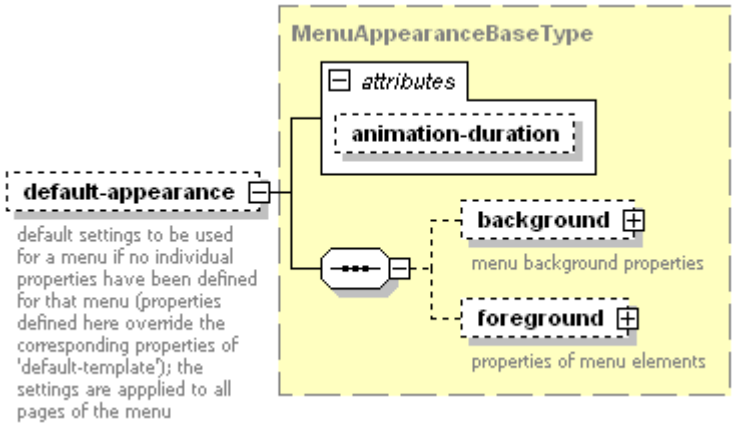
Name	Type	Use	Default	Fixed
default-thumbnail	**derived by: xs:string**	optional		
first-play-title-id	**xs:IDREF**	optional		
title-finish-action	**derived by: xs:string**	optional	PlayNextTitle	

8.16 element DiscProjectBaseType/menu/default-template

The optional default-template will be used when no template element is provided for the individual menu elements.

diagram	
type	extension of **MenuTemplateBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
attributes	Name / Type / Use / Default / Fixed id / **xs:ID** / optional src / **FileLocation** / optional group / **xs:string** / optional name / **derived by: xs:string** / required

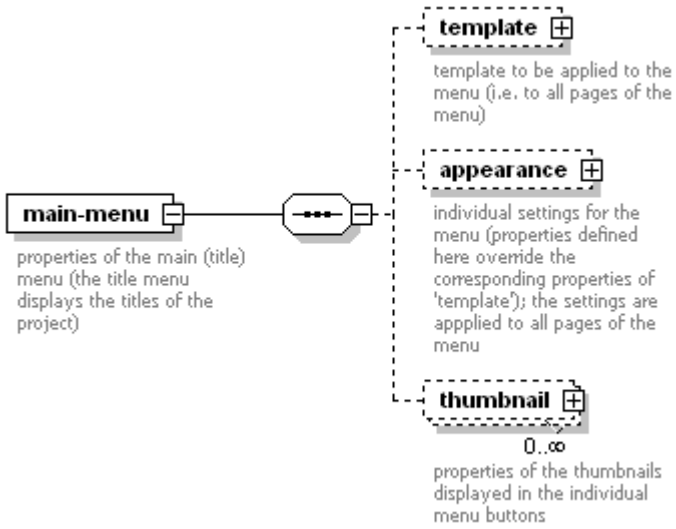
8.17 element DiscProjectBaseType/menu/default-appearance

diagram	
type	**MenuAppearanceBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
children	**background foreground**

attributes	Name	Type	Use	Default	Fixed
	animation-duration	**AVTime**	optional		

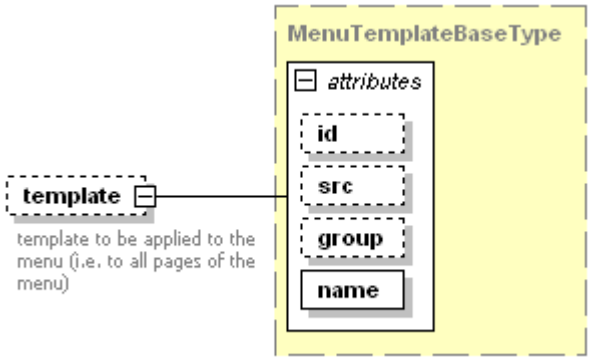
8.18 element DiscProjectBaseType/menu/main-menu

Typically, a main-menu is used when there is more than one video title or a combination of video titles and slide shows.

diagram	
properties	isRef 0 content complex
children	**template appearance thumbnail**

8.19 element DiscProjectBaseType/menu/main-menu/template

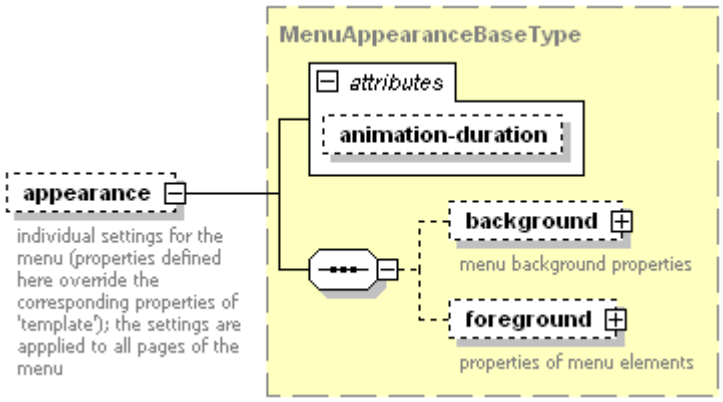
The optional template for the main menu.

diagram	
type	**MenuTemplateBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex

attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	optional		
	group	**xs:string**	optional		
	name	**derived by: xs:string**	required		

8.20 element DiscProjectBaseType/menu/main-menu/appearance

diagram



type extension of **MenuAppearanceBaseType**

properties

isRef	0
minOcc	0
maxOcc	1
content	complex

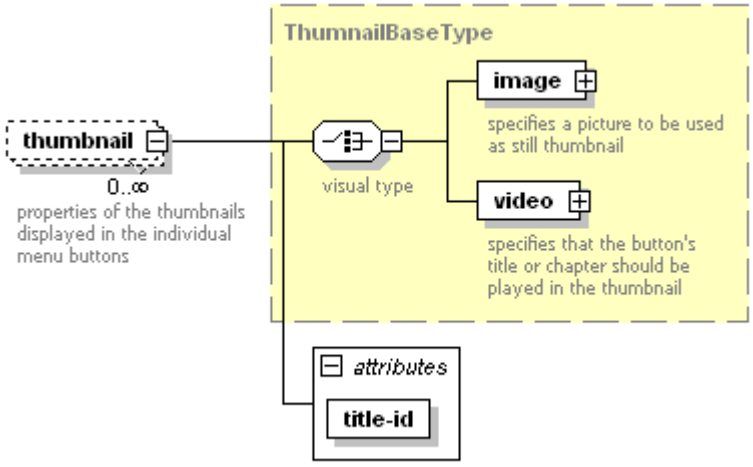
children **background foreground**

attributes	Name	Type	Use	Default	Fixed
	animation-duration	**AVTime**	optional		

8.21 element DiscProjectBaseType/menu/main-menu/thumbnail

A representation for a video chapter or slide show in the main menu.

diagram

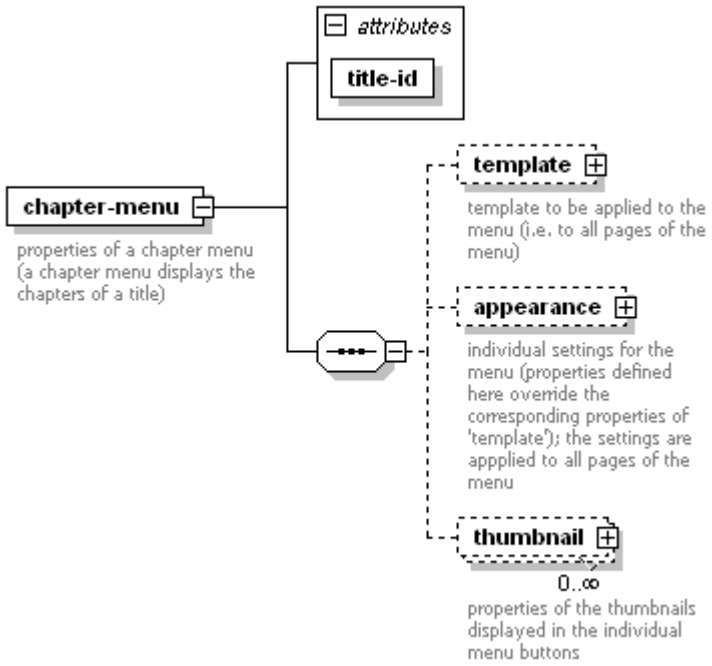


type extension of **ThumnailBaseType**

properties	isRef 0 minOcc 0 maxOcc unbounded content complex				
children	**image video**				
attributes	Name	Type	Use	Default	Fixed
	title-id	**xs:IDREF**	required		

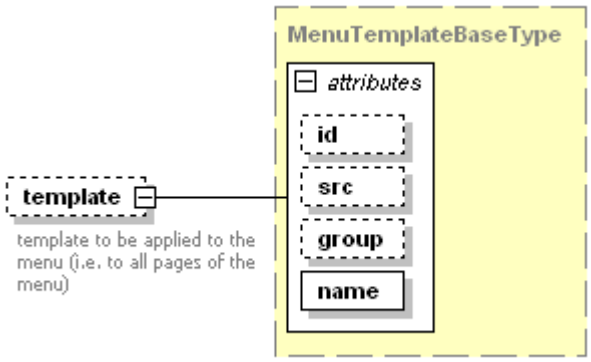
8.22 element chapter-menu

The chapter-menu element represents a single page that links to individual chapters of a video title. The chapter-menu can have a template of its own and a number of thumbnails.

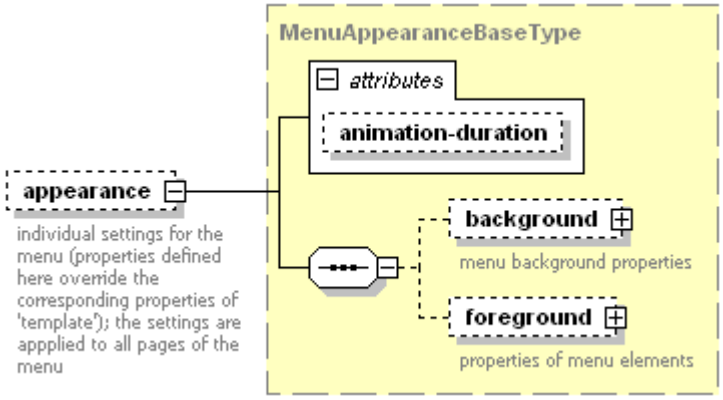
diagram					
properties	content complex				
children	**template appearance thumbnail**				
used by	elements **DiscProjectBaseType/menu DiscProjectBaseType/menu**				
attributes	Name	Type	Use	Default	Fixed
	title-id	**xs:IDREF**	required		

8.23 element chapter-menu/template

The template element of chapter-menu represents the background and button style.

diagram	
type	extension of **MenuTemplateBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
attributes	Name / Type / Use / Default / Fixed id / **xs:ID** / optional src / **FileLocation** / optional group / **xs:string** / optional name / **derived by: xs:string** / required

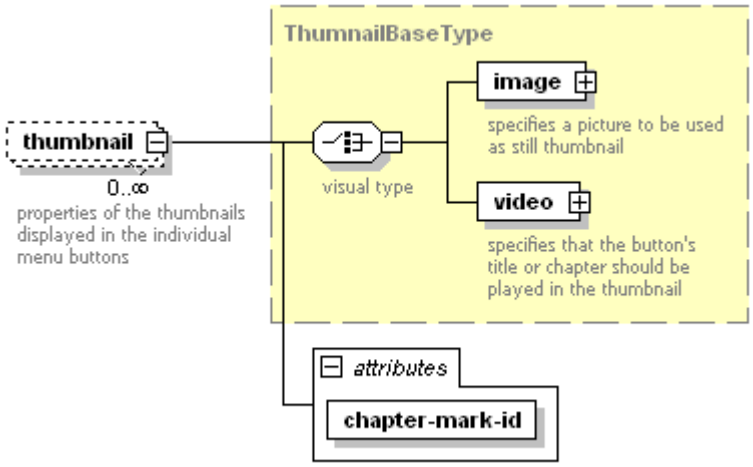
8.24 element chapter-menu/appearance

diagram	
type	**MenuAppearanceBaseType**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
children	**background foreground**
attributes	Name / Type / Use / Default / Fixed animation-duration / **AVTime** / optional

8.25 element chapter-menu/thumbnail

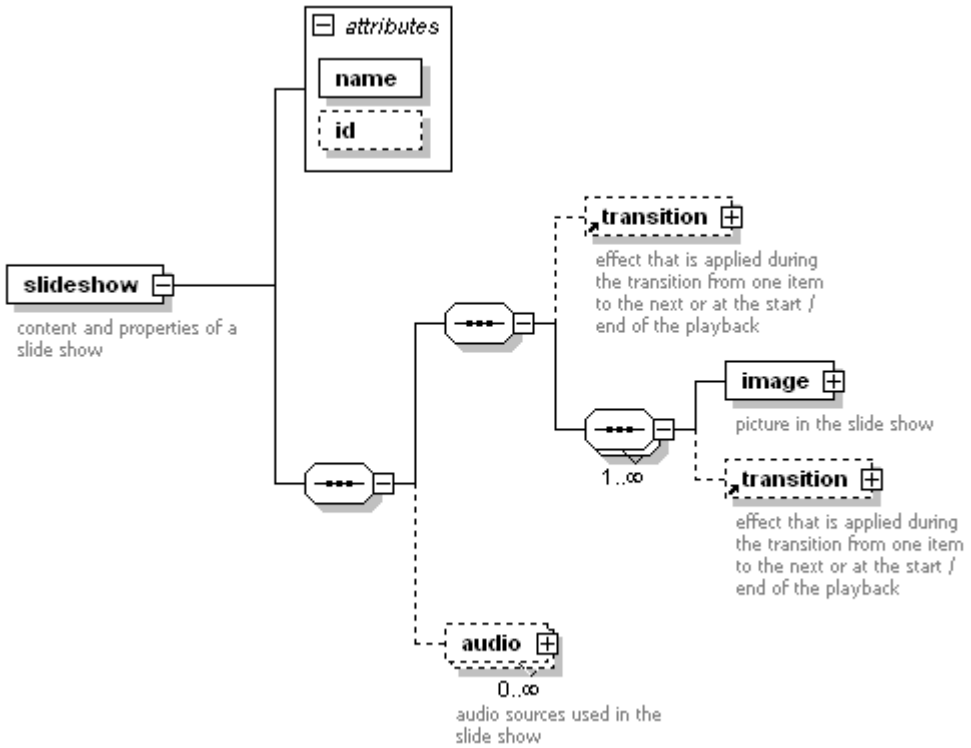
The thumbnail element of chapter-menu represents a small image or video to facilitate orientation for the user. The source of the thumbnail can either be a picture file or a video title.

The thumbnail must link to an existing chapter-id.

diagram	
type	extension of **ThumnailBaseType**
properties	isRef 0 minOcc 0 maxOcc unbounded content complex
children	**image video**
attributes	Name: chapter-mark-id; Type: **xs:IDREF**; Use: required; Default: ; Fixed:

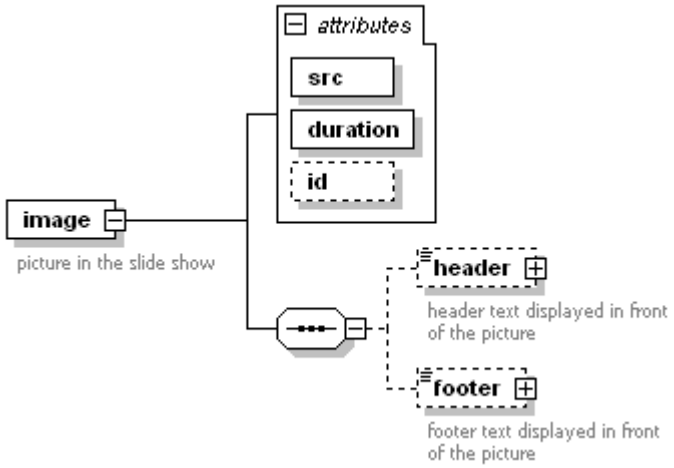
8.26 element slideshow

A slideshow is a sequence of images and optional transitions between them.

diagram	
properties	content complex
children	**transition image audio**
used by	element **DiscProjectBaseType/content**
attributes	Name / Type / Use / Default / Fixed name / **xs:string** / required id / **xs:ID** / optional

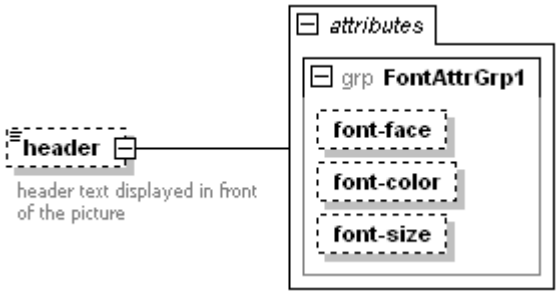
8.27 element slideshow/image

Individual element for slide shows. Every image is displayed for the time given in the "duration" attribute.

diagram	
properties	isRef 0 content complex
children	**header footer**
attributes	Name / Type / Use / Default / Fixed src / **FileLocation** / required duration / **AVTime** / required id / **xs:ID** / optional

8.28 element slideshow/image/header

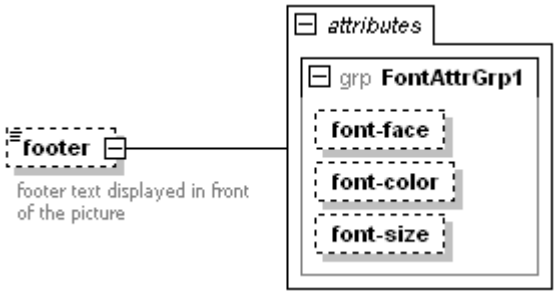
An optional header text for an image.

diagram	
type	extension of **xs:string**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
attributes	Name / Type / Use / Default / Fixed font-face / **derived by: xs:string** / optional font-color / **RGBColor** / optional font-size / **derived by: xs:int** / optional

8.29 element slideshow/image/footer

An optional footer text for an image.

diagram



type: extension of **xs:string**

properties

isRef	0
minOcc	0
maxOcc	1
content	complex

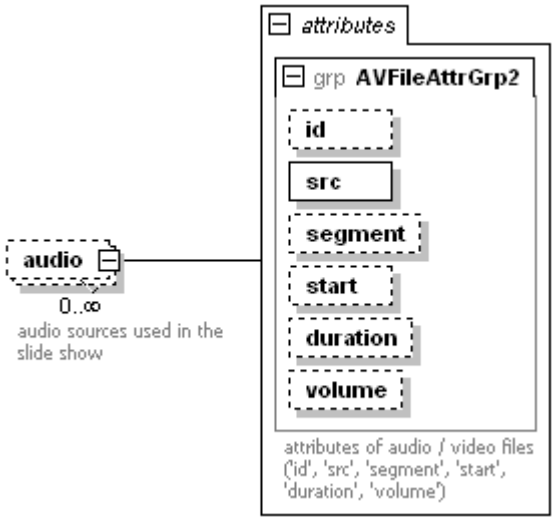
attributes

Name	Type	Use	Default	Fixed
font-face	**derived by: xs:string**	optional		
font-color	**RGBColor**	optional		
font-size	**derived by: xs:int**	optional		

8.30 element slideshow/audio

Slideshows can have several optional audio tracks.

diagram



properties

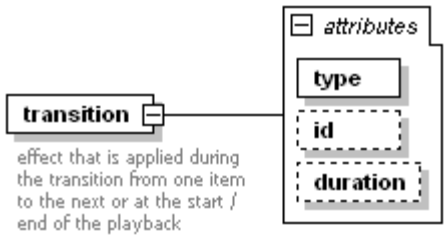
isRef	0
minOcc	0
maxOcc	unbounded
content	complex

attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	required		
	segment	**derived by: xs:int**	optional		
	start	**AVTime**	optional	0	
	duration	**AVTime**	optional		
	volume	**Percentage**	optional	100%	

8.31 element transition

NeroVision API optionally allows separating the individual pictures in a slideshow by transitions. The transitions determine how images appear and disappear, e.g. with a wiping effect.

diagram



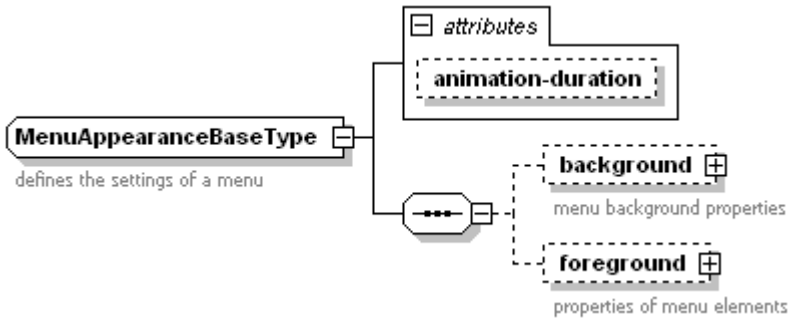
properties content complex

used by elements **slideshow slideshow**

attributes	Name	Type	Use	Default	Fixed
	type	**EffectType**	required		
	id	**xs:ID**	optional		
	duration	**AVTime**	optional	1.5	

8.32 complexType MenuAppearanceBaseType

diagram

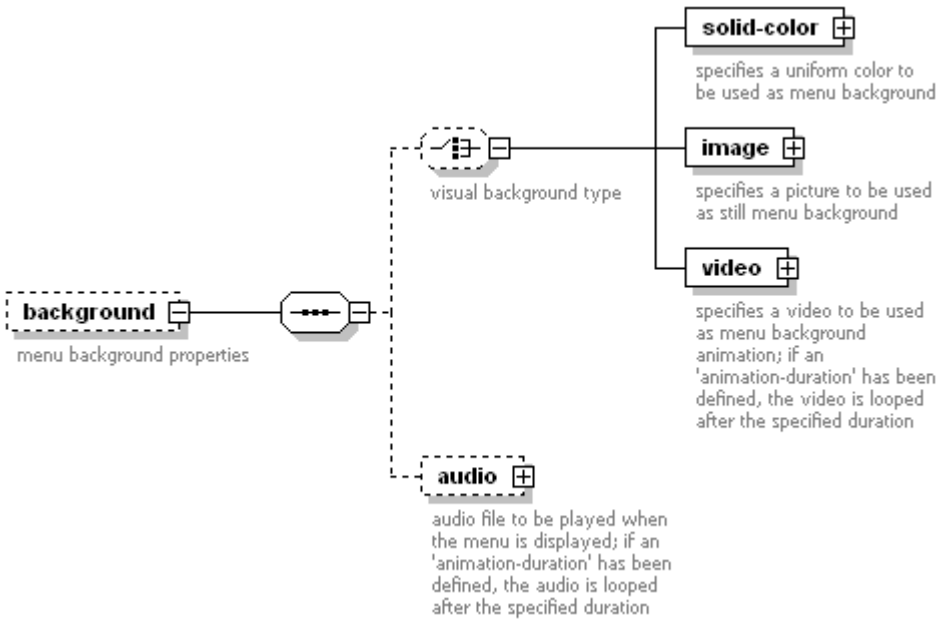


children **background foreground**

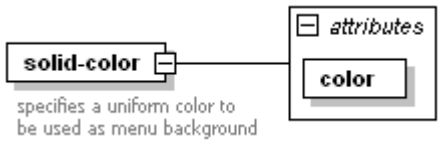
used by elements **chapter-menu/appearance DiscProjectBaseType/menu/main-menu/appearance DiscProjectBaseType/menu/default-appearance**

attributes	Name	Type	Use	Default	Fixed
	animation-duration	**AVTime**	optional		

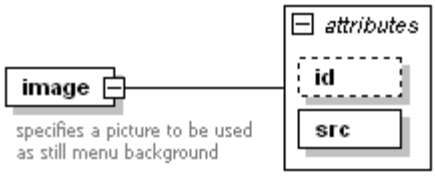
8.33 element MenuAppearanceBaseType/background

diagram	
properties	isRef 0 minOcc 0 maxOcc 1 content complex
children	**solid-color image video audio**

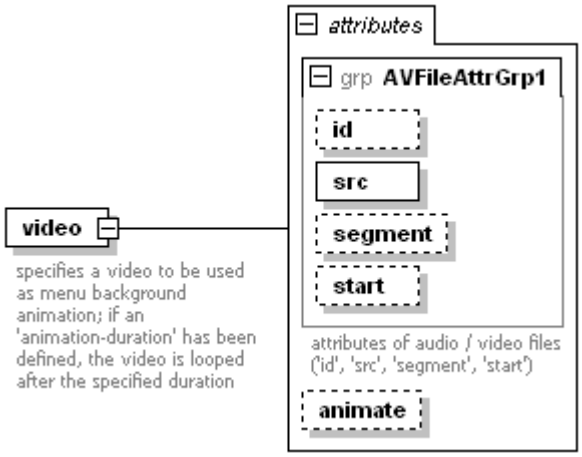
8.34 element MenuAppearanceBaseType/background/solid-color

diagram	
properties	isRef 0 content complex
attributes	Name: color; Type: **RGBColor**; Use: required; Default: ; Fixed:

8.35 element MenuAppearanceBaseType/background/image

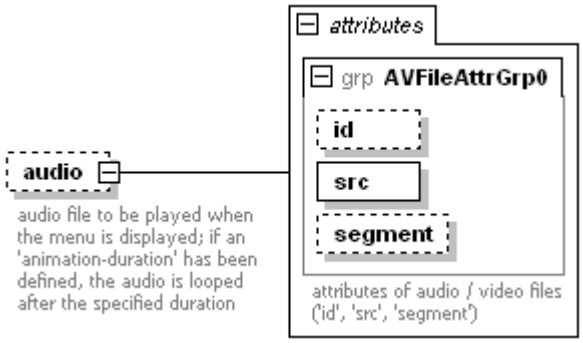
diagram	
properties	isRef 0 content complex
attributes	Name / Type / Use / Default / Fixed id / **xs:ID** / optional src / **FileLocation** / required

8.36 element MenuAppearanceBaseType/background/video

diagram	
properties	isRef 0 content complex
attributes	Name / Type / Use / Default / Fixed id / **xs:ID** / optional src / **FileLocation** / required segment / **derived by: xs:int** / optional start / **AVTime** / optional / 0 animate / **xs:boolean** / optional / true

8.37 element MenuAppearanceBaseType/background/audio

diagram



properties

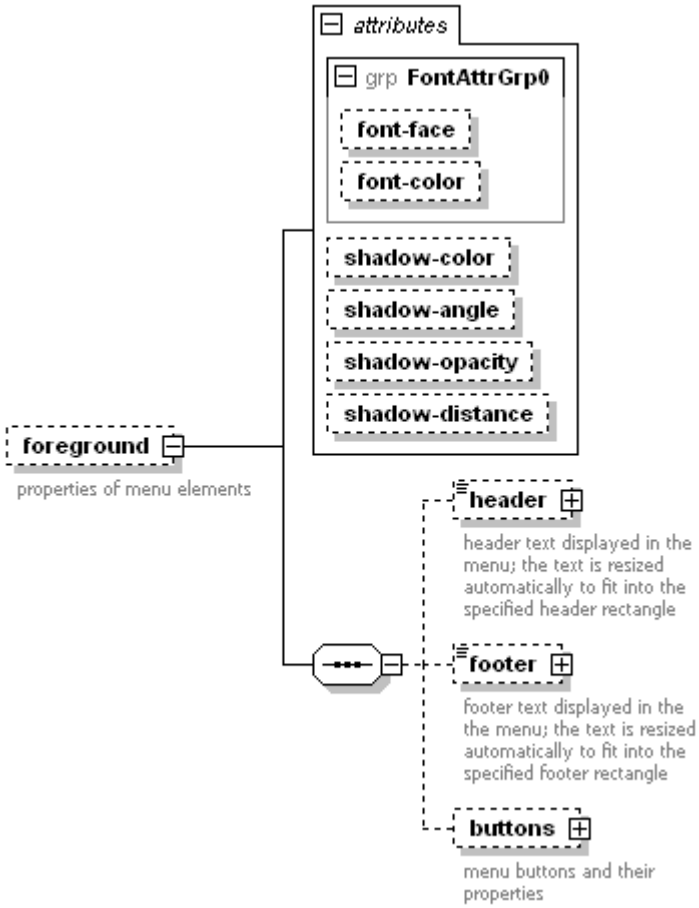
isRef	0
minOcc	0
maxOcc	1
content	complex

attributes

Name	Type	Use	Default	Fixed
id	**xs:ID**	optional		
src	**FileLocation**	required		
segment	**derived by: xs:int**	optional		

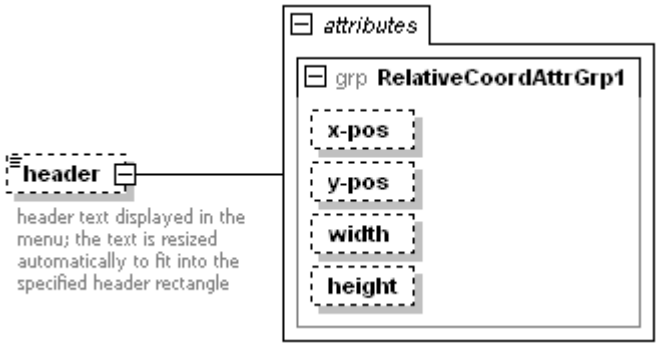
8.38 element MenuAppearanceBaseType/foreground

diagram

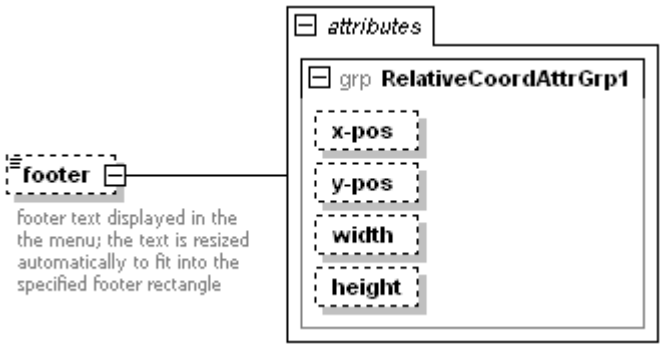


properties	isRef 0 minOcc 0 maxOcc 1 content complex
children	**header footer buttons**
attributes	Name / Type / Use / Default / Fixed font-face / **derived by: xs:string** / optional font-color / **RGBColor** / optional shadow-color / **RGBColor** / optional shadow-angle / **derived by: xs:int** / optional shadow-opacity / **Percentage** / optional shadow-distance / **derived by: xs:int** / optional

8.39 element MenuAppearanceBaseType/foreground/header

diagram	
type	extension of **xs:string**
properties	isRef 0 minOcc 0 maxOcc 1 content complex
attributes	Name / Type / Use / Default / Fixed x-pos / **Percentage** / optional y-pos / **Percentage** / optional width / **Percentage** / optional height / **Percentage** / optional

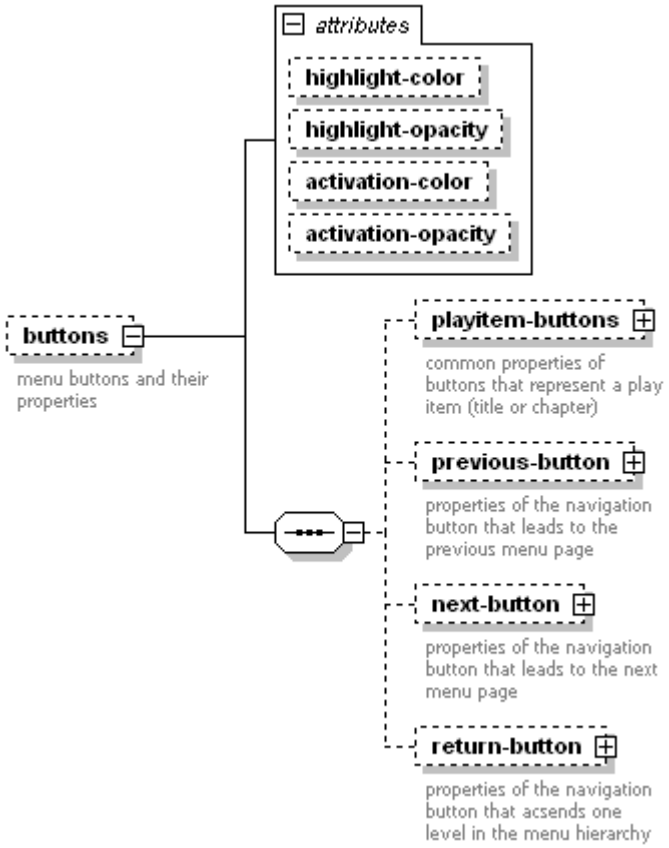
8.40 element MenuAppearanceBaseType/foreground/footer

diagram	
type	extension of **xs:string**

properties	isRef	0			
	minOcc	0			
	maxOcc	1			
	content	complex			
attributes	Name	Type	Use	Default	Fixed
	x-pos	**Percentage**	optional		
	y-pos	**Percentage**	optional		
	width	**Percentage**	optional		
	height	**Percentage**	optional		

8.41 element MenuAppearanceBaseType/foreground/buttons

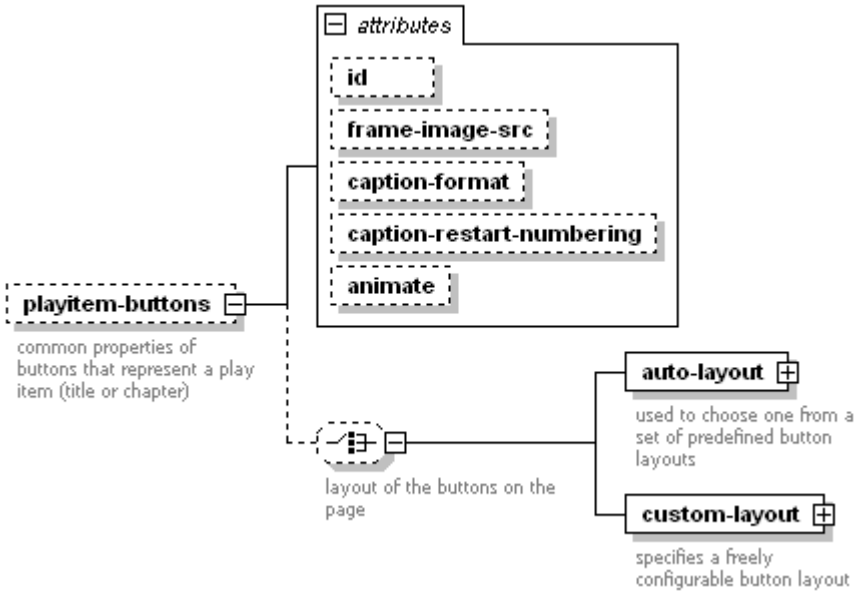
diagram



properties	isRef	0			
	minOcc	0			
	maxOcc	1			
	content	complex			
children	**playitem-buttons previous-button next-button return-button**				
attributes	Name	Type	Use	Default	Fixed
	highlight-color	**RGBColor**	optional		
	highlight-opacity	**Percentage**	optional		
	activation-color	**RGBColor**	optional		
	activation-opacity	**Percentage**	optional		

8.42 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons

diagram



properties

isRef	0
minOcc	0
maxOcc	1
content	complex

children

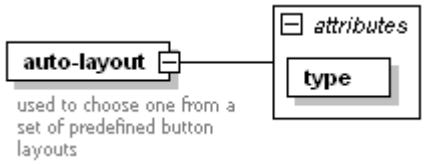
auto-layout custom-layout

attributes

Name	Type	Use	Default	Fixed
id	**xs:ID**	optional		
frame-image-src	**FileLocation**	optional		
caption-format	**derived by: xs:string**	optional		
caption-restart-numbering	**xs:boolean**	optional		
animate	**xs:boolean**	optional	false	

8.43 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/auto-layout

diagram



properties

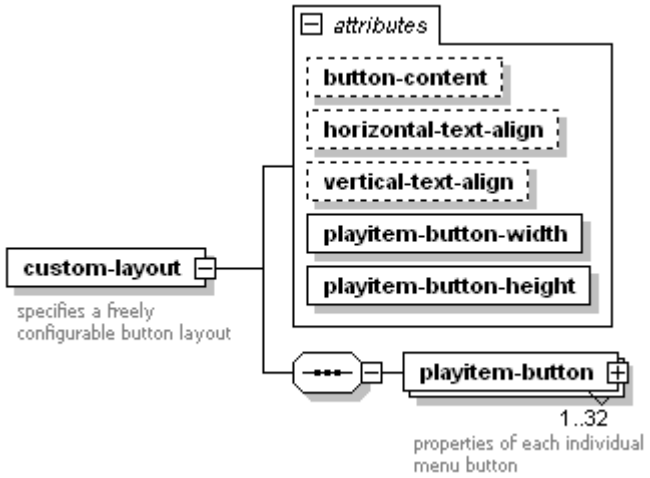
isRef	0
content	complex

attributes

Name	Type	Use	Default	Fixed
type	**derived by: xs:string**	required		

8.44 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout

diagram



properties

isRef	0
content	complex

children

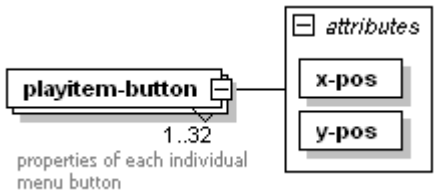
playitem-button

attributes

Name	Type	Use	Default	Fixed
button-content	**derived by: xs:string**	optional	TextAndThumbnail	
horizontal-text-align	**derived by: xs:string**	optional	Left	
vertical-text-align	**derived by: xs:string**	optional	Top	
playitem-button-width	**Percentage**	required		
playitem-button-height	**Percentage**	required		

8.45 element MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/playitem-button

diagram



properties

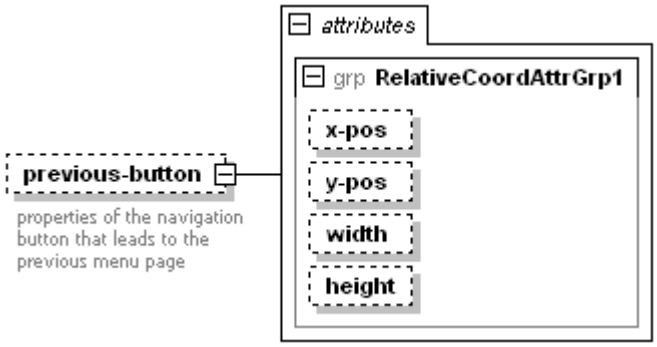
isRef	0
minOcc	1
maxOcc	32
content	complex

attributes

Name	Type	Use	Default	Fixed
x-pos	**Percentage**	required		
y-pos	**Percentage**	required		

8.46 element MenuAppearanceBaseType/foreground/buttons/previous-button

diagram



properties

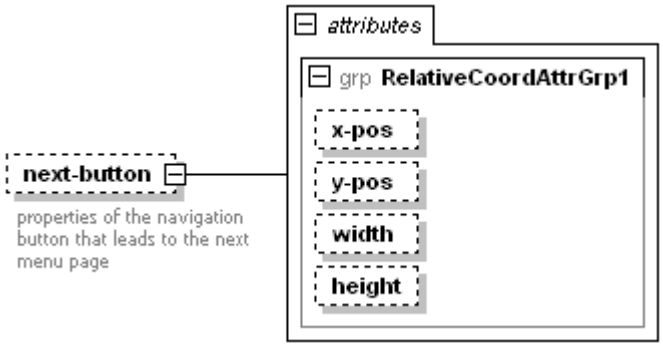
isRef	0
minOcc	0
maxOcc	1
content	complex

attributes

Name	Type	Use	Default	Fixed
x-pos	**Percentage**	optional		
y-pos	**Percentage**	optional		
width	**Percentage**	optional		
height	**Percentage**	optional		

8.47 element MenuAppearanceBaseType/foreground/buttons/next-button

diagram



properties

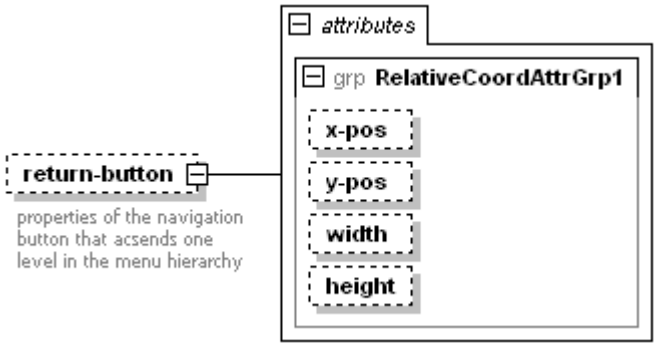
isRef	0
minOcc	0
maxOcc	1
content	complex

attributes

Name	Type	Use	Default	Fixed
x-pos	**Percentage**	optional		
y-pos	**Percentage**	optional		
width	**Percentage**	optional		
height	**Percentage**	optional		

8.48 element MenuAppearanceBaseType/foreground/buttons/return-button

diagram



properties

isRef	0
minOcc	0
maxOcc	1
content	complex

attributes

Name	Type	Use	Default	Fixed
x-pos	**Percentage**	optional		
y-pos	**Percentage**	optional		
width	**Percentage**	optional		
height	**Percentage**	optional		

8.49 complexType MenuTemplateBaseType

This type represents a *NeroVision 2* menu template.

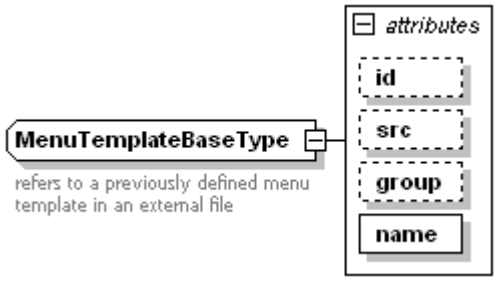
If "src" has been provided, this will be interpreted as a file location. Without "src" the standard *NeroVision 2* templates (if installed) will be used.

"group" and "name" are the optional group and required name of the template to be used from the "src" template file.

If no group information has been provided, the first template with matching name will be used.

If the requested template does not exist, *NeroVision API* will return an error. The error object will contain the XML-ID of the offending XML element - if the optional ID was provided beforehand.

diagram



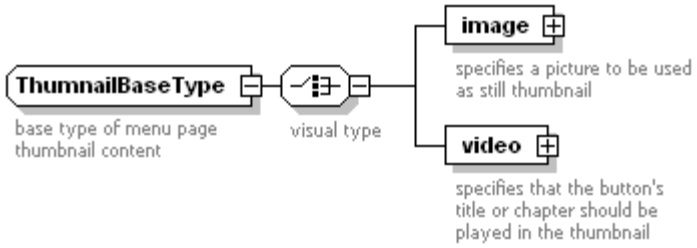
used by

elements **DiscProjectBaseType/menu/default-template chapter-menu/template**
DiscProjectBaseType/menu/main-menu/template

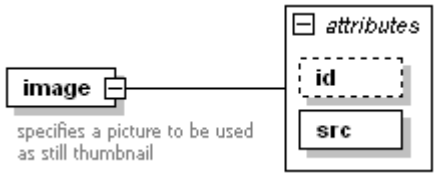
attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	optional		
	group	**xs:string**	optional		
	name	**derived by: xs:string**	required		

8.50 complexType ThumnailBaseType

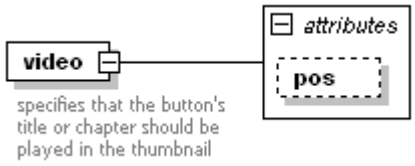
A thumbnail represents a chapter and can be created from an image file or a video file.

diagram	
children	**image video**
used by	elements **chapter-menu/thumbnail DiscProjectBaseType/menu/main-menu/thumbnail**

8.51 element ThumnailBaseType/image

diagram	
properties	isRef 0 content complex
attributes	Name / Type / Use / Default / Fixed id / **xs:ID** / optional src / **FileLocation** / required

8.52 element ThumnailBaseType/video

diagram	
properties	**isRef 0** **content complex**
attributes	**Name / Type / Use / Default / Fixed** **pos / derived by: AVTime / optional**

8.53 simpleType AVTime

Time measured in seconds. This type is used for durations as well as relative positions in video titles.

type	restriction of **xs:decimal**
used by	element **ThumnailBaseType/video** attributes **MenuAppearanceBaseType/@animation-duration transition/ @duration slideshow/image/@duration AVFileAttrGrp2/ @duration videotitle/chapters/chapter-mark/@pos AVFileAttrGrp1/@start**
facets	minInclusive 0

8.54 simpleType FileLocation

The location of a video or image.

type	**xs:anyURI**
used by	attributes **MenuAppearanceBaseType/foreground/buttons/playitem-buttons/@frame-image-src slideshow/image/@src MenuTemplateBaseType/@src MenuAppearanceBaseType/background/image/@src ThumnailBaseType/image/@src AVFileAttrGrp0/@src**

8.55 simpleType Percentage

Used to set the volume in percent for audio tracks in slideshows.

type	restriction of **xs:string**
used by	attributes **MenuAppearanceBaseType/foreground/buttons/@activation-opacity RelativeCoordAttrGrp1/ @height MenuAppearanceBaseType/foreground/buttons/@highlight-opacity MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/ @playitem-button-height** **MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/ @playitem-button-width MenuAppearanceBaseType/foreground/@shadow-opacity AVFileAttrGrp2/@volume RelativeCoordAttrGrp1/@width MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/playitem-button/ @x-pos RelativeCoordAttrGrp0/@x-pos MenuAppearanceBaseType/foreground/buttons/playitem-buttons/custom-layout/playitem-button/@y-pos RelativeCoordAttrGrp0/@y-pos**
facets	pattern \d+(\.\d+)?%

8.56 simpleType RGBColor

Used to determine font color for headers and footers.

The intensity of the red, green and blue channels (RGB) is given in hexadecimal numbers, formatted as #<red><green><blue>.

For example, #FFFFFF is white, #FF0000 red or #000000 black.

type	restriction of **xs:string**
used by	attributes **MenuAppearanceBaseType/foreground/buttons/@activation-color MenuAppearanceBaseType/background/solid-color/@color FontAttrGrp0/@font-color MenuAppearanceBaseType/foreground/buttons/@highlight-color MenuAppearanceBaseType/foreground/@shadow-color**

facets	pattern #[0-9A-Fa-f]{6}

8.57 simpleType VideoEncoding

This is a region-dependent setting for the main television systems with the most significant difference being the frame rate.

PAL (Phase Alternating Line) produces 25 frames per second and NTSC (National Television System Committee) approximately 30 frames per second.

The resolution also differs slightly.

type	restriction of **xs:string**
used by	attribute **DiscProjectBaseType/@encoding**
facets	enumeration PAL enumeration NTSC

8.58 simpleType EffectType

The following transition values are built-in for compatibility with previous versions of *NeroVision API*.

- WipeUp
- WipeLeft
- WipeRight
- WipeDown
- MoveIn
- MoveOut

Apart from those, effects are identified by a GUID. NeroVision API provides methods to enumerate all available video, text and transition effects. Thus, the GUID of an individual effect can be determined and used in the XML description file.
An additional method is available for translating the effect to one of the supported languages. The application that uses NeroVision API can therefore display the effect names in any desired language and reference the effects language-independently through their GUIDs.

type	union of (restriction of **xs:string**, restriction of **xs:string**)
used by	attribute **transition/@type**

8.59 attributeGroup AVFileAttrGrp0

used by	element	**MenuAppearanceBaseType/background/audio**			
	attributeGroup	**AVFileAttrGrp1**			
attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	required		
	segment	**derived by: xs:int**	optional		

8.60 attributeGroup AVFileAttrGrp1

used by	element	MenuAppearanceBaseType/background/video			
	attributeGroup	AVFileAttrGrp2			
attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	required		
	segment	**derived by: xs:int**	optional		
	start	**AVTime**	optional	0	

8.61 attributeGroup AVFileAttrGrp2

used by	elements	videotitle/audio slideshow/audio videotitle/video			
attributes	Name	Type	Use	Default	Fixed
	id	**xs:ID**	optional		
	src	**FileLocation**	required		
	segment	**derived by: xs:int**	optional		
	start	**AVTime**	optional	0	
	duration	**AVTime**	optional		
	volume	**Percentage**	optional	100%	

Description of attributes	
src	This required attribute points to the actual file. The full path must be given.
segment	A (DVB-) MPEG file can contain multiple logical MPEG segments. A segment roughly correlates with an individual film. When such a (DVB-) MPEG file is being scanned by the MediaAnalyzer, a number of segments will be found. The relevant methods are • IFileInfo::NumSegments() • IFileInfo::GetSegmentMediaInfo() The available segments can be used independently in a project, e.g. if only one film in the DVB file is of interest. The attribute “segment” (0..n) is used to identify the portions. Please note that this attribute is zero-based. So, for example, segment=”2” points to the third segment. For files that contain non-segmented MPEG or no MPEG at all, this attribute can be omitted or set to zero (first segment).
start	All content before the position that is identified by “start” will be ignored. The unit for “start” is seconds.
duration	Content will only be considered for as many seconds as “duration” specifies. This attribute can be combined with “start”.
volume	The unit for volume is percent. The volume of the media file can be reduced for normalization purposes.
id	IDs can be provided by the user to uniquely identify items.

8.62 attributeGroup FontAttrGrp0

used by	element	**MenuAppearanceBaseType/foreground**			
	attributeGroup	**FontAttrGrp1**			
attributes	Name	Type	Use	Default	Fixed
	font-face	**derived by: xs:string**	optional		
	font-color	**RGBColor**	optional		

8.63 attributeGroup FontAttrGrp1

used by	elements	**slideshow/image/footer slideshow/image/header**			
attributes	Name	Type	Use	Default	Fixed
	font-face	**derived by: xs:string**	optional		
	font-color	**RGBColor**	optional		
	font-size	**derived by: xs:int**	optional		

8.64 attributeGroup RelativeCoordAttrGrp0

used by	attributeGroup	**RelativeCoordAttrGrp1**			
attributes	Name	Type	Use	Default	Fixed
	x-pos	**Percentage**	optional		
	y-pos	**Percentage**	optional		

8.65 attributeGroup RelativeCoordAttrGrp1

used by	elements	**MenuAppearanceBaseType/foreground/footer MenuAppearanceBaseType/foreground/header MenuAppearanceBaseType/foreground/buttons/next-button MenuAppearanceBaseType/foreground/buttons/previous-button MenuAppearanceBaseType/foreground/buttons/return-button**			
attributes	Name	Type	Use	Default	Fixed
	x-pos	**Percentage**	optional		
	y-pos	**Percentage**	optional		
	width	**Percentage**	optional		
	height	**Percentage**	optional		

9 FAQ

This is a collection of frequently asked questions from the *NeroSDK* forum and Nero AG's customer support.

How must the XML definition file be changed to allow audio for a slide show?

```
<slideshow>
   <image src=... />
   ...
   <audio src="c:\winnt\media\tada.wav" id="gc1"/>
   <audio src="c:\winnt\media\tada2.wav" volume="50%" start="1"
                                                    duration="2" id="gc2"/>
</slideshow>
```

How do I create or edit a menu template?
Currently, you have to use *NeroVision* to create new menu templates.
User-defined templates are saved under
"<UserApplicationData>\Ahead\NeroVision\nve-mtmpl.bin".
Support for defining menu templates in XML might be added for later releases.

In my (S)VCD, the DVD navigate arrow keys have no function. I have to hit the "1" key to start the first menu entry. What's going on here?
You created a (S)VCD, not a DVD.
VCD/SVCDs navigation always uses the numeric keys 0-9 and the next/prev/return/play keys. This behaviour complies with the (S)VCD standard.

I would like to use a custom picture for the background of the menu screen, but how do I define the backgound JPEG or BMP without it getting defined in the nve-mtmpl.bin as an absolute path?

See the answer to the next question.

I would like to define a template similar to the sample "Island" template where the actual source image is stored relative to the template (e.g. in the same directory or a "Pictures" sub-directory)?

Your picture file has to be stored beneath *NeroVision's* installation path when editing the template. The installation path will be stripped from the picture file path when saving the menu template. When reloading it, the current *NeroVision* installation path is prepended to the picture file path. Preferrably you would put the picture file into

```
%InstallPath%\MenuTemplates\Pictures\pic.jpg
```

while the menu template is

```
%InstallPath%\MenuTemplates\template1.bin
```

(You can use any file title for menu template files.)

Is there a way to make the menu thumbnail something other than the first slide, e.g. <menu default-thumbnail="FirstNonEmptyFrame">?

Not individually per thumbnail, but globally for all menus:

```
<menu default-thumbnail="FirstFrame">
```

This will result in the use of the first frame (for all thumbnails in the menu).

```
<menu default-thumbnail="FirstNonEmptyFrame">
```

This will result in the use of the first non empty frame (for all thumbnails in themenu).

I have content that is DVD-conforming. Still, transcoding takes a long time. Which video file format must be referenced in the XML so that the *NeroVision API* will not transcode the file?

The *NeroVision API* will only **smart encode** a video if the the video files were analyzed by the „MediaAnalyzer“ object before the (S)VCD or DVD is created through the use of the CreateNeroBurnContext function.
The reason is that determining conformity can take a very long time and render uniform progress during the execution of CreateNeroBurnContext impossible.

Apart from that, the MPEG profile has to match the target profile. Therein, the same restrictions that apply for *NeroVision* are in effect.

10 Known Limitations

The limitations of the *NeroVision API* correspond to those of the *NeroAPI*:

- Currently there are no Linux versions of the *NeroVision API* and the *Nero* Software Development Kit (*NeroSDK*).
- Methods of the *NeroAPI* and *NeroVision API* should not be called simultaneously from different threads.
- Only one recorder can be accessed at a time.