Video

Plug-in User Guide

2.2.1.165787


vizrt

Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied.

This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Last Updated

06.04.2015


vizrt

Table of Contents

1 Introduction 7
1.1 Supported Workflows 8
1.1.1 Manual Workflows 8
1.1.2 Automated Workflows 10
2 Installation 13
2.1 Conventions 13
2.2 Install the Video Plug-in 13
2.3 Verify The Installation 15
2.4 Update The Database Schema 15
3 Configuration 17
3.1 Configure The Video Plug-in 17
3.1.1 TranscodingConfiguration.properties 18
3.1.2 Configuration.properties 20
3.1.3 Creating Specialized Transcoding Configurations 21
3.1.4 Setting the VME Online Group Parameters 22
3.2 Define Media Content Types 22
3.2.1 Defining an External Video Content Type 23
3.2.2 Defining an Internal Video Content Type 24
3.2.3 Defining an External Audio Content Type 25
3.2.4 Defining an Internal Audio Content Type 26
3.2.5 Adding a Group Selection Field 27
3.2.6 Including Key Frames in Video Content Items 28
3.3 Configure Viz Content Pilot 29
3.4 Define a Proxy Service for VME Online 29
3.5 Configure VME Production 30
3.5.1 Define a Named Service for VME 30
3.5.2 Define a Proxy Authentication Service for VME 31
3.5.3 Enable CORS on the VME Server 31
3.6 Advanced Configuration for VME Online 32
3.6.1 Improved Transcoding Control 32
4 Using the Content Studio Plug-in 35
4.1 Adding the Viz Content Pilot Panel 35
4.2 Working With Media Content Items 36
4.2.1 Creating a Video Content Item 36

4.2.2 Creating an Audio Content Item 41
4.2.3 Publishing Media Content Items 42
5 Automated Media Publishing 45
5.1 Web Service-Based Automation 45
5.2 Syndication-Based Automation 46
5.3 Working With the Video/Audio Field 47
5.3.1 Specifying Basic Data 48
5.3.2 Specifying Crop Points 48
5.3.3 Specifying Timeline Information 48
5.3.4 VME Online Status URI 50
6 Accessing Media from Templates 53

1 Introduction

The Escenic Content Engine's Video plug-in supports integration of the Content Engine with **Viz Media Engine (VME)**, making it possible to:

- Access media (video/audio) content archived in a Viz Media Engine from Escenic web sites
- Use Viz Media Engine to:
  - transcode media content
  - generate **key frames** from video content
  - send transcoded media content to the required location (a content delivery network for example)

  The source media content may be stored in a number of different locations:

  - A Viz Media Engine
  - In some other archive elsewhere on the net
  - Locally, in a location known to the Content Engine

The plug-in also incorporates a Content Studio extension that enables Content Studio users to easily:

- Access and browse media content stored in a Viz Media Engine
- Decorate selected video content by adding template-based graphics (for example, lead in and lead out graphics, graphic overlays and so on).
- Add selected and decorated video content to suitably-configured content items
- Crop videos before publishing (that is, select a playback start point and end point so that only a segment of the video is published).
- Add cue points to videos (significant points from which playback can be started).
- Add transient links to videos (links that appear and disappear while the video is playing)
- Use Viz Media Engine to transcode media content.

Some of this extra Content Studio functionality is only **fully** available if:

- You are running Content Studio on Microsoft Windows
- You have a correctly configured Viz Content Pilot **Newsroom Component** (from version 5.6.1) installed on your computer

Specifically, if you do not have Viz Content Pilot installed on your computer or are using a different operating system, then:

- You will not be able to access Viz Media Engine to select media content from **within** Content Studio. You will, however, still be able to add media from Viz Media Engine by drag-and-drop from **Viz Media Studio**.
- You will not be able to add template-based graphics to videos.

Most of the extra functionality that the Video plug-in provides access to is provided by Vizrt video components: Viz Content Pilot, and Viz Media Engine. This functionality is therefore not described in any detail here: if you need to know more then you should consult the documentation for the relevant product.

1.1 Supported Workflows

The purpose of the Video plug-in is to enable the Content Engine to be used together with various other Vizrt systems to provide efficient production lines for online publishing of video content. All the systems involved are large and complex and can be configured in many different ways. The systems can therefore theoretically be connected together in many different way. In practice, however, this is not the case: not all configurations will work, or work well. This section contains descriptions of configurations and workflows that are known to work satisfactorily and are therefore supported by Vizrt.

All the described workflows require access to a **VME Online** system, which performs transcoding and generation of key frame images.

1.1.1 Manual Workflows

These workflows are driven from Content Studio: the Content Studio user creates a video or audio content item and then adds a video or audio clip either from Viz Media Engine or from his/her local machine.

These workflows are described in detail (from the Content Studio user's point of view) in **chapter 4**.

1.1.1.1 VME and VCP-based Workflow

In addition to **VME Online**, this workflow requires:

- Access to a **VME Video Production** system for storage/archiving of media content. Earlier **Viz Ardome** systems are also supported (version 4.8.3 or later)
- Access to a **Viz Content Pilot** system
- That Content Studio is being accessed from Microsoft Windows systems
- That Viz Content Pilot's **Newsroom Component** is installed on the computers used to access Content Studio

Publishing media in this workflow consists of the following steps:

1. Content Studio user creates a video or audio content item.
2. Content Studio user selects a video or audio clip stored in VME, and adds it to the content item.
3. For a video clip, the Content Studio user makes any required modifications using the timeline editor.
4. Content Studio user saves the content item.
5. The Content Engine sends the location of the included video or audio clip in VME to the VME Online system.

6. VME Online retrieves the video or audio clip from VME.
7. VME Online transcodes the retrieved video or audio clip into the required web formats.
8. VME Online updates the transcoding status of the content item in the Content Engine.
9. VME Online sends the transcoded video/audio clips to the required location (a content delivery network, for example).
10. VME Online returns the URLs of the transcoded clips, key frames and so on to the Content Engine.
11. The Content Engine updates the content item with the returned data.

Steps 1 and 2 in the above workflow can also be in reverse order: if you select a video or audio clip and drag it into Content Studio, a suitable content item is automatically created to hold it.

1.1.1.2 VME-based Workflow

In addition to **VME Online**, this workflow requires:

- Access to a **VME Video Production** system for storage/archiving of video/audio content. Earlier **Viz Ardome** systems are also supported (version 4.8.3 or later)
- Access to **Viz Media Studio** (the VME Video Production web interface)

Publishing media in this workflow consists of the following steps:

1. Content Studio user creates a video or audio content item.
2. Content Studio user selects a video or audio clip from Viz Media Studio and adds it to the content item by drag-and-drop.
3. For a video clip, the Content Studio user makes any required modifications using the timeline editor.
4. Content Studio user saves the content item.
5. The Content Engine sends the location of the included video or audio clip in VME to the VME Online system.
6. VME Online retrieves the video or audio clip from VME.
7. VME Online transcodes the retrieved video or audio clip into the required web formats.
8. VME Online updates the transcoding status of the content item in the Content Engine.
9. VME Online sends the transcoded video/audio clips to the required location (a content delivery network, for example).
10. VME Online returns the URLs of the transcoded clips, key frames and so on to the Content Engine.
11. The Content Engine updates the content item with the returned data.

Steps 1 and 2 in the above workflow can also be in reverse order: if you select a video or audio clip and drag it into Content Studio, a suitable content item is automatically created to hold it.

1.1.1.3 Local Upload Workflow

This workflow is supported for both video and audio. It only requires access to a **VME Online** system. Note that this workflow is not recommended for large-scale use, since the Content Engine is not optimized for managing large volumes of media data. It is primarily intended for small scale, casual use.

Publishing media in this workflow consists of the following steps:

1. Content Studio user creates a video or audio content item.
2. Content Studio user uploads a video or audio clip from his/her local machine.
3. The Content Engine sends the URL of the uploaded video or audio clip to the VME Online system
4. VME Online retrieves the video or audio clip from the Content Engine.
5. VME Online transcodes the retrieved video or audio clip into the required web formats.
6. VME Online updates the transcoding status of the content item in the Content Engine.
7. VME Online sends the transcoded video/audio clips to the required location (a content delivery network, for example).
8. VME Online returns the URLs of the transcoded clips, key frames and so on to the Content Engine.
9. The Content Engine updates the content item with the returned data.

1.1.2 Automated Workflows

These workflows are supported for both video and audio. These are driven by some external system (for example, another content management system) and do not require any interaction from Content Studio users. There are two ways of creating an automated workflow:

- Using the Content Engine web service
- Using the Content Engine syndication subsystem

The automated workflows described here are based on the assumption that the media clips to be published are stored either in a VME Video Production system or in some other external system. There is no description of how to create an automated version of the local upload workflow described in **section 1.1.1.3**. This workflow is only intended for small-scale casual use and is not regarded as suitable for automation.

1.1.2.1 Web Service-based Workflow

In a web-service based workflow, the external system that is driving the publishing process communicates with the Content Engine via the Content Engine's web service, effectively emulating Content Studio.

Publishing media in this workflow consists of the following steps:

1. The external system obtains the URL of the video or audio content to be published. The content may be stored either in a **VME Video Production**

system or in some other archiving system. The only requirement is that the media archiving system makes the video/audio clips accessible to the **VME Online** system in some way. It might, for example, publish them on an HTTP, FTP or SMB server.

2. The external system creates a suitable content item in the Content Engine, using the Content Engine's web service. The created content item contains the URL of the video or audio clip to be published.
3. The Content Engine sends the URL of the included video or audio clip to VME Online.
4. VME Online retrieves the video/audio clip from the specified location.
5. VME Online transcodes the retrieved video or audio clip into the required web formats.
6. VME Online updates the transcoding status of the content item in the Content Engine.
7. VME Online sends the transcoded video/audio clips to the required location (a content delivery network, for example).
8. VME Online returns the URLs of the transcoded clips, key frames and so on to the Content Engine.
9. The Content Engine updates the content item with the returned data.

1.1.2.2 Syndication-based Workflow

In a syndication-based workflow, the external system that is driving the publishing process communicates with the Content Engine by generating an Escenic syndication format file and starting an import process.

Publishing media in this workflow consists of the following steps:

1. The external system obtains the URL of the video or audio content to be published. The content may be stored either in a **VME Video Production** system or in some other archiving system. The only requirement is that the media archiving system makes the video/audio clips accessible to the **VME Online** system in some way. It might, for example, publish them on an HTTP, FTP or SMB server.
2. The external system creates an Escenic syndication format file that defines one or more video or audio content items. Each content item definition contains the URL of the video or audio clip to be published.
3. The external system saves the syndication file in the Content Engine's import folder.
4. The Content Engine imports the syndication file, creating the content items defined in the syndication file.
5. The Content Engine sends the URL of the included video or audio clip to VME Online
6. VME Online retrieves the video/audio clip from the specified location.
7. VME Online transcodes the retrieved video or audio clip into the required web formats.
8. VME Online updates the transcoding status of the content item in the Content Engine.

9. VME Online sends the transcoded video/audio clips to the required location (a content delivery network, for example).
10. VME Online returns the URLs of the transcoded clips, key frames and so on to the Content Engine.
11. The Content Engine updates the content item with the returned data.

2 Installation

The following preconditions must be met before you can install Video 2.2.1.165787 in a Content Engine:

- Escenic Content Engine version 5.6 or higher and Escenic assembly tool have been installed as described in the **Escenic Content Engine Installation Guide** and are in working order.
- You have the required plug-in distribution file **`video-dist-2.2.1.165787.zip`**.

2.1 Conventions

The instructions in the following section assume that you have a standard Content Engine installation, as described in the **Escenic Content Engine Installation Guide** for releases 5.6 and above. *escenic-home* is used to refer to the **`/opt/escenic`** folder under which both the Content Engine itself and all plug-ins are installed).

The Content Engine and the software it depends on may be installed on one or several host machines depending on the type of installation required. In order to unambiguously identify the machines on which various installation actions must be carried out, the **Escenic Content Engine Installation Guide** defines a set of special host names that are used throughout the manual.

Some of these names are also used here:

assembly-host
: The machine used to assemble the various Content Engine components into a enterprise archive or .EAR file.

engine-host
: The machine(s) used to host application servers and Content Engine instances.

The host names always appear in a bold typeface. If you are installing everything on one host you can, of course, ignore them: you can just do everything on the same machine. If you are creating a larger multi-host installation, then they should help ensure that you do things in the right places.

2.2 Install the Video Plug-in

Installing the Video plug-in involves the following steps:

1. Log in as **`escenic`** on your **assembly-host**.
2. Download the Video distribution from the Escenic Technet web site (http://technet.escenic.com). If you have a multi-host installation with shared folders as described in the **Escenic Content Engine Installation**

Guide, then it is a good idea to download the distribution to your shared **`/mnt/download`** folder:

```
$ cd /mnt/download
$ wget http://user:password@technet.escenic.com/downloads/56/video-dist-2.2.1.165787.zip
```

Otherwise, download it to some temporary location of your choice.

3. The Video plug-in depends on another plug-in called **VCP Editor** that manages the integration with Viz Content Pilot's Newsroom Component. You therefore need to download this plug-in's distribution file to the same location:

   ```
   $ wget http://user:password@technet.escenic.com/downloads/56/vcpeditor-dist-n.n.zip
   ```

   (Download whichever **VCP Editor** version is currently recommended for your version of the Content Engine.)

4. If the folder **`/opt/escenic/engine/plugins`** does not already exist, create it:

   ```
   $ mkdir /opt/escenic/engine/plugins
   ```

5. Unpack the downloaded distribution files:

   ```
   $ cd /opt/escenic/engine/plugins
   $ unzip /mnt/download/video-dist-2.2.1.165787.zip
   $ unzip /mnt/download/vcpeditor-dist-n.n.zip
   ```

   - **`/opt/escenic/engine/plugins/video`**
   - **`/opt/escenic/engine/plugins/vcpeditor`**

6. Run the **`ece`** script to re-assemble your Content Engine applications

   ```
   $ ece assemble
   ```

   This generates an EAR file (**`/var/cache/escenic/engine.ear`**) that you can deploy on all your **engine-host**s.

7. ---

   If you have a single-host installation, then skip this step.

   On each **engine-host**, copy **`/var/cache/escenic/engine.ear`** from the **assembly-host**. If you have installed an SSH server on the **`assembly-host`** and SSH clients on your **`engine-host`**s, then you can do this as follows:

   ```
   $ scp -r escenic@assembly-host-ip-address:/var/cache/escenic/engine.ear /var/cache/escenic/
   ```

   where *assembly-host-ip-address* is the host name or IP address of your **assembly-host**.

8. On each **engine-host**, deploy the EAR file and restart the Content Engine by entering:

   ```
   $ ece deploy
   $ ece restart
   ```

2.3 Verify The Installation

To verify the status of the Video plug-in, open the Escenic Admin web application (usually located at **`http://`***server*`/escenic-admin`**) and click on **View installed plugins**. The status of all currently installed plug-ins is shown here, and indicated as follows:

The plug-in is correctly installed.

The plug-in is not correctly installed.

If the Video plug-in is correctly installed, you should see something like this in the displayed plug-in list:

2.4 Update The Database Schema

The Video plug-in needs some additions to be made to the Content Engine database schema. The scripts needed to make the required additions are included in the **`misc/database/`** folder of the distribution. There are two sets of scripts, one for MySql databases, in **`misc/database/mysql`**, and one for Oracle databases in **`misc/database/oracle`**. There are three scripts in each folder:

- **`constraints.sql`**
- **`indexes.sql`**
- **`tables.sql`**

To run the scripts:

1. Log in as **`escenic`** on your **database-host**.
2. Copy or unpack the appropriate scripts for your database to an appropriate location (for example **`/tmp/video/misc/database/mysql`**).
3. Run the scripts as follows

   - For MySql:

     ```
     $ cd /tmp/video/misc/database/mysql/
     $ for el in tables.sql indexes.sql constraints.sql; do \
       mysql -u ece-user -pece-password -h dbhost db-name < $el
       done;
     ```

     replacing *db-name*, *dbhost*, *ece-user* and *ece-password* with the correct values for your database.

3 Configuration

In order to be able to use the Video plug-in after installing it (see **chapter 2**) you must:

1. Create a configuration file containing all the information the Video plug-in needs to access the various systems involved in your video workflows.
2. Add one or more video and audio content types to your publication's **`content-type`** resource.
3. Configure Viz Content Pilot correctly to ensure that the Content Studio browse functionality works correctly (only necessary if you are intending to use the VME and VCP-based workflow described in **section 1.1.1.1**).
4. Define a **proxy service** for the VME Online system that you will use for transcoding.
5. Configure the VME Production system in which media content is archived. This is only necessary if you are intending to use one of VME-based workflows. If you are only using the local upload workflow, or if your media is stored in an external system other than VME, then this step is not required.

In addition to the above steps, you can also (if required) control the handling of video and audio in different sections of your publications by adding section parameters.

While you are working on the configuration, it is a good idea to set the Content Engine logging level to **`DEBUG`** for the video plug-in. This will ensure that you get detailed information in the log file. Once the system is correctly configured you can set the logging level back to the default level. To change the logging level open a browser and go the **`escenic-admin`** application's **Logging Levels** page. Set the category **`com.escenic.video`** to the level you require (for example, **`DEBUG`**). For general information about setting logging levels, see the **Escenic Content Engine Server Administration Guide**.

3.1 Configure The Video Plug-in

The Video plug-in needs information about how to access the VME Online system responsible for transcoding. You provide this information by adding one or more configuration files to one of your configuration layers.

The configuration files you need to add are:

`TranscodingConfig.properties`
: This configuration file contains all the information needed to manage the transcoding process. Normally you will only need to create one copy of this file, but if you want to have different transcoding set-ups for different publications, then you will need to create several copies of it with different names (for example **`TranscodingConfigPub1.properties`**, **`TranscodingConfigPub2.properties`** and so on.

`Configuration.properties`
You only need to create one of these files if:

- Your system is required to support a local upload workflow (see**section 1.1.1.3**) and/or
- You want to have different transcoding set-ups for different publications (see **section 3.1.3**)

3.1.1 TranscodingConfiguration.properties

This configuration file contains all the information needed to manage the transcoding process.

1. Create the configuration file by copying **`/opt/escenic/engine/plugins/video/config/engine/com/escenic/video/TranscodingConfig.properties`** to your common configuration layer. For example:

```
$ mkdir -p /etc/escenic/engine/common/com/escenic/video/
$ cp /opt/escenic/engine/plugins/video/config/engine/com/escenic/video/
TranscodingConfig.properties \
    /etc/escenic/engine/common/com/escenic/video/TranscodingConfig.properties
```

2. Open your configuration file for editing.
3. Define values for some or all of the properties in the file. See the property descriptions below for details.

The following properties may be defined in **`TranscodingConfiguration.properties`**. The descriptions specify whether or not a setting is required.

`webServiceUri` (required)
The URI of your Content Engine web service, including a valid user name and password. For example:

```
webServiceUri=http://user-name:password@host/webservice/
```

where:

- *user-name* is the name of an Escenic user with access to all video content items
- *password* is the password of this user
- *host* is the Content Engine's host name or IP address

Any special characters in the URI must be URI-encoded.

This URI is used by the VME Online system when transcoding to access video content items in the Content Engine. VME Online needs full read/write access to these video content items. Before transcoding, it reads the URI of the actual video content to be transcoded plus any meta-information it needs from a video content item. During transcoding it updates the content item with status information. When transcoding is complete it updates the content item with the URIs of the published transcoded versions of the video, key frames and so on.

`adactusServiceUri` (required)
The URI of your VME Online service, including a valid user name and password. For example:

```
adactusServiceUri=http://user-name:password@host/
```

where:

- *user-name* is the name of a VME Online user
- *password* is the password of this user
- *host* is the VME Online service's host name or IP address

Any special characters in the URI must be URI-encoded. If, for example, the user name contains an @ character, this must be escaped as follows:

```
adactusServiceUri=http://post%40my-company.com:very_secret@10.211.10.8/
```

`provider` (required)
The name of the VME Online **provider** that is to own the transcoded videos. The named provider must be defined in the VME Online system. For further information about what a VME Online provider is, see the **Viz Media Engine Administrator's Guide**.

`group.video` (required)
The name of the VME Online **group** to which the transcoded videos are to belong. The named group must:

- Be defined in the VME Online system.
- Belong to the specified **`provider`**.
- Have a default **publish point**.
- Have the **Auto Transcode** option set to **`true`**.
- Have the **Auto Transcode and Publish** option set to **`false`**.

For further information about what a VME Online group is, see the**Viz Media Engine Administrator's Guide**.

`group.audio` (required)
The name of the VME Online **group** to which the transcoded audios are to belong. The named group must:

- Be defined in the VME Online system.
- Belong to the specified **`provider`**.
- Have a default **publish point**.
- Have the **Auto Transcode** option set to **`true`**.
- Have the **Auto Transcode and Publish** option set to **`false`**.

For further information about what a VME Online group is, see the**Viz Media Engine Administrator's Guide**.

`adactusXslFile` (optional)
This parameter does not normally need to be set. If set it must point to an XSL transformation file that will be applied to video content items when they are retrieved by VME Online. It can be used to apply modifications to the content items before they are submitted to VME

Online. A sample XSL file is supplied with the plug-in and can be found here:

```
adactusXslFile=/opt/escenic/plugins/video/misc/example/add-adaptation.xsl
```

If you have special requirements, you can use this file as a starting point.

`externalReferencePrefix` (optional)
A prefix used by VME Online when accessing the Content Engine. The default value is **`escenic:`**, and if the VME online system is only required to interact with one Escenic system, then you do not need to set this property. If, however, you have several parallel Escenic installations (a production system and a test or staging system, for example), all of which use the same VME Online system, then you must set this property to a different value in each Escenic installation. For example:

```
externalReferencePrefix=escenic-staging:
```

3.1.2 Configuration.properties

This configuration file contains optional settings that are only required if:

- Your system is required to support a local upload workflow (see**section 1.1.1.3**) and/or
- You want to have different transcoding set-ups (that is, different versions of **`TranscodingConfiguration.properties`**) for different publications (see **section 3.1.3**)

To create a **`Configuration.properties`** file

1. Copy **`/opt/escenic/engine/plugins/video/config/engine/com/escenic/video/Configuration.properties`** to your common configuration layer. For example:

```
$ mkdir -p /etc/escenic/engine/common/com/escenic/video/
$ cp /opt/escenic/engine/plugins/video/config/engine/com/escenic/video/Configuration.properties
 \
    /etc/escenic/engine/common/com/escenic/video/Configuration.properties
```

2. Open your configuration file for editing.
3. Define values for some or all of the properties in the file. See the property descriptions below for details.

The following properties may be defined in **`Configuration.properties`**:

`binaryPrefix`
This property is only required if your system is required to support a local upload workflow (see**section 1.1.1.3**). Any video or audio clips uploaded to the Content Engine must be made accessible to the VME Online server by means of HTTP, FTP, SMB or some other network protocol. So before you can set this property you must make sure that an appropriate server is installed for serving the files to VME Online. You can then set this property to the appropriate URI.

If, for example you have set up an HTTP server to provide access to the files, you would need to specify:

```
binaryPrefix=http://host-name/
```

where *host-name* is the host name or IP address of the server you have set up.

If your system is not required to support local upload of videos, then you do not need to set this property.

`publicationTranscodeMapping.`*pub-name*

If you need to set up different transcoding configurations for different publications, then you can specify several such properties, where *pub-name* is the name of a publication that requires specialized transcoding.

The value of each property must be the name of a configuration file containing the specialized transcoding configuration, for example:

```
publicationTranscodeMapping.DailyNews=./TranscodingConfigNews.properties
publicationTranscodeMapping.WeeklyGossip=./TranscodingConfigGlossy.properties
publicationTranscodeMapping.Entertainment=./TranscodingConfigGlossy.properties
```

Note that, as in the example above, you can share a configuration file between several publications.

You do not need to specify an entry for all your publications. If a publication is not specified here, then the default transcoding configuration specified in **`TranscodingConfig.properties`** is used.

3.1.3 Creating Specialized Transcoding Configurations

If you have set **`publicationTranscodeMapping.`***pub-name* properties in a **`Configuration.properties`** file, then you need to create the specialized **`TranscodingConfig.properties`** files that they reference. To do this:

1. Make a copy of your default **`TranscodingConfiguration.properties`** file and rename it to whatever name you have chosen.
2. Open it for editing.
3. Change the property settings as required.
4. Save your changes.

When you are editing this file, bear in mind the following points:

- A specialized transcoding configuration file **must have** a

  ```
  $class=com.escenic.video.TranscodingConfig
  ```

  declaration at the top of the file. (This line is optional in the main **`TranscodingConfig.properties`** file.)
- You can set properties to use a value set in one of the other transcoding configuration files. To use a value set in the main **`TranscodingConfig.properties`** file, for example you can specify something like this:

  ```
  adactusServiceUri=${./TranscodingConfig.adactusServiceUri}
  ```

3.1.4 Setting the VME Online Group Parameters

The VME Online **`group.video`** and **`group.audio`** parameters are very important parameter settings for both video and audio content items since it determines how they are transcoded, where they are published and quality of service settings. In order to provide maximum flexibility, the Video plug-in allows you to set them in a number of different places, and at different levels. Settings made at lower levels override the more global settings made at higher levels.

The parameter can be set at four different levels:

Global level
The settings made with **`group.video`** and **`group.audio`** in **`TranscodingConfiguration.properties`** (see **section 3.1.1**) are global default settings.

Publication level
You can override the global **`group.video`** or **`group.audio`** setting for a specific publication or group of publications by setting the parameter in a specialized transcoding configuration file that you reference from a **`Configuration.properties`** file as described in **section 3.1.2**.

Section level
You can override the global or publication level **`group.video`** or **`group.audio`** setting for part of a publication by setting a section parameter in one of the publication's section. The section parameters to use have the following names:

- **`com.vizrt.group.video`** for video content
- **`com.vizrt.group.audio`** for audio content

As is always the case with section parameters, these parameters apply to the section in which they are set and all of its subsections.

Content item level
You can also allow Content Studio users to override group settings from individual video and audio content items by including a group selection field your video and audio content type definitions. For a description of how do do this, see **section 3.2.5**.

3.2 Define Media Content Types

In order to be able to use the Video plug-in, you need to add at least one suitably configured media content type to your publication's **`content-type`** resource. For general information about the **`content-type`** resource and how to edit it, see the **Escenic Content Engine Resource Reference**.

A media content type must either be a **video** content type or an **audio** content type, and both of them can either be **external** or **internal**:

External
An external video/audio content type can be used to hold metadata referencing content stored either in a Viz Media Engine or in some other external server. External content types are used for all workflows **except** the local upload workflow described in **section 1.1.1.3**.

Internal
An internal video/audio content type includes a **`link`** field for holding an actual video/audio object (an MPEG or MP3 file, for example). Such content items can therefore hold video/audio content, not just metadata and a reference. Internal content types are used for the local upload workflow described in **section 1.1.1.3**.

The Video Plug-in distribution file contains an example **`content-type`** resource file (**`video/misc/example/content-type.xml`**) with more complete examples of the content type definitions described in the following sections.

3.2.1 Defining an External Video Content Type

A video content type must at least have the following:

- A parameter called **`com.vizrt.video`** that is set to **`true`**
- A parameter called **`com.escenic.article.staging`** that is set to **`false`**
- A **`basic`** field with:
  - **`mime-type`** set to **`application/json`**
  - a **`video`** sub-element with a **`value`** attribute set to **`true`**. This element must belong to the namespace **`http://xmlns.escenic.com/2010/video`**.
- A decorator called **`videoArticleDecorator`**

The following example shows such a minimal **`content-type`**:

```
<content-type name="external-video">
  <parameter name="com.vizrt.video" value="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="video" type="basic" mime-type="application/json">
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
    </field>
  </panel>
</content-type>
```

Here is a more realistic version of the same example, this time including a title field, label and so on, but with the important parts highlighted.

```
<content-type name="external-video">
  <parameter name="com.vizrt.video" value="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>External video</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="video" type="basic" mime-type="application/json">
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
    </field>
  </panel>
```

```
</content-type>
```

The **com.vizrt.video** parameter identifies the content type as a video content type managed by the Video plug-in. The **com.escenic.article.staging** parameter specifies that **content item staging** must be disabled for this content type. If content item staging is disabled generally at your installation or for the whole publication, then you can omit this parameter. (For general information about content item staging, see the **Escenic Content Engine Advanced Developer Guide**.)

On Windows clients only, and only if Vizrt Content Pilot is installed, Content Studio includes a special Viz Content Pilot panel that you use to select video from Viz Media Engine. For more information about this, see **section 4.2**.

You can use external video content types to include video content from any external source, not just from Viz Media Engine. However, only Viz Media Engine video content can be directly accessed from within Content Studio. Video content from other sources can only be added via the web service (see **section 5.1**) or the import system (see **section 5.2**).

3.2.2 Defining an Internal Video Content Type

An internal video content type must have all the same fields, decorators and parameters as an external video type, but must in addition have a **link** field for holding a link to a locally stored video file. The following example shows such a content-type, with the additional **link** field highlighted:

```
<content-type name="internal-video">
  <parameter name="com.vizrt.video" value="true"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>Internal video</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="binary" type="link">
      <relation>com.escenic.edit-media</relation>
    </field>
    <field name="video" type="basic" mime-type="application/json">
      <ui:hidden/>
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
    </field>
  </panel>
</content-type>
```

When a new content item of this type is created in Content Studio, an **Open file** dialog is automatically displayed so that the user can upload a suitable file to the **link** field. You can use a **constraints** element to limit the file types it is possible to upload. For example:

```
<field name="binary" type="link">
  <constraints>
    <mime-type>video/mpeg</mime-type>
    <mime-type>video/mp4</mime-type>
  </constraints>
</field>
```

You can add **mime-type** elements specifying any video data formats supported by VME Online. For a complete list of supported formats, see the VME Online documentation.

3.2.3 Defining an External Audio Content Type

An audio content type must at least have the following:

- A parameter called **com.vizrt.audio** that is set to **true**
- A parameter called **com.escenic.article.staging** that is set to **false**
- A **basic** field with:
  - **mime-type** set to **application/json**
  - an **audio** sub-element with a **value** attribute set to **true**. This element must belong to the namespace **http://xmlns.escenic.com/2013/audio**.
- A decorator called **audioArticleDecorator**

The following example shows such a minimal **content-type**:

```
<content-type name="external-audio">
  <parameter name="com.vizrt.audio" value="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:decorator name="audioArticleDecorator"/>
  <panel name="main">
    <field name="audio" type="basic" mime-type="application/json">
      <ui:hidden/>
      <audio xmlns="http://xmlns.escenic.com/2013/audio" enabled="true"/>
    </field>
  </panel>
</content-type>
```

Here is a more realistic version of the same example, this time including a title field, label and so on, but with the important parts highlighted.

```
<content-type name="external-audio">
  <parameter name="com.vizrt.audio" value="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>External audio</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="audioArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="audio" type="basic" mime-type="application/json">
      <ui:hidden/>
      <audio xmlns="http://xmlns.escenic.com/2013/audio" enabled="true"/>
    </field>
  </panel>
</content-type>
```

The **com.vizrt.audio** parameter identifies the content type as an audio content type managed by the Video plug-in. The **com.escenic.article.staging** parameter specifies that **content item staging** must be disabled for this content type. If content item staging is disabled generally at your installation or for the whole publication, then you can omit this parameter. (For general information about content item staging, see the **Escenic Content Engine Advanced Developer Guide**.)

You can use external audio content types to include audio content from any external source. Audio content from these external sources can only

be added via the web service (see **section 5.1**) or the import system (see **section 5.2**).

3.2.4 Defining an Internal Audio Content Type

A internal audio content type must at least have the following:

- A parameter called **com.vizrt.audio** that is set to **true**
- A parameter called **com.escenic.article.staging** that is set to **false**
- A **link** field for holding a link to a locally stored audio file
- A **basic** field with:
  - **mime-type** set to **application/json**
  - an **audio** sub-element with a **value** attribute set to **true**. This element must belong to the namespace **http://xmlns.escenic.com/2013/audio**.
- A decorator called **audioArticleDecorator**

The following example shows such a minimal **content-type**:

```
<content-type name="internal-audio">
  <parameter name="com.vizrt.audio" value="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>Internal audio</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="audioArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="binary" type="link">
      <relation>com.escenic.edit-media</relation>
    </field>
    <field name="audio" type="basic" mime-type="application/json">
      <ui:hidden/>
      <audio xmlns="http://xmlns.escenic.com/2013/audio" enabled="true"/>
    </field>
  </panel>
</content-type>
```

When a new content item of this type is created in Content Studio, an **Open file** dialog is automatically displayed so that the user can upload a suitable file to the **link** field. You can use a **constraints** element to limit the file types it is possible to upload. For example:

```
<field name="binary" type="link">
  <constraints>
    <mime-type>audio/mp3</mime-type>
    <mime-type>audio/mp4</mime-type>
  </constraints>
</field>
```

The **com.vizrt.audio** parameter identifies the content type as an audio content type managed by the Video plug-in. The **com.escenic.article.staging** parameter specifies that **content item staging** must be disabled for this content type. If content item staging is disabled generally at your installation or for the whole publication, then you can omit this parameter. (For general information about content item staging, see the **Escenic Content Engine Advanced Developer Guide**.)

You can add **mime-type** elements specifying any audio data formats supported by VME Online. For a complete list of supported formats, see the VME Online documentation.

3.2.5 Adding a Group Selection Field

By adding a **group selection field** to your video and audio content types, you can allow Content Studio users to override the default VME Online group setting that determines how audio and video is transcoded and published. A group selection field must have its **mime-type** set to **text/plain** and must have this element as one of its children:

```
<group xmlns="http://xmlns.escenic.com/2013/media" enabled="true"/>
```

Note that the element **must** belong to the namespace **http://xmlns.escenic.com/2013/media**. Here is a very simple example of a group selection field:

```
<field name="vmeogroup" type="basic" mime-type="text/plain">
  <ui:label/>Transcoding Group</ui:label/>
  <group xmlns="http://xmlns.escenic.com/2013/media" enabled="true"/>
</field>
```

Adding this definition to you video content type will cause a **Transcoding Group** field to appear in Content Studio when users edit video content items. The user can then enter the name of group he wants to be used when the video is processed by VME Online. However, this means that the user needs to know the names of the VME Online groups, and enter them correctly. A better solution would be to get the names of all the available groups from VME Online, and allow the user to select the required group from a list.

You can do this by using a collection field instead of a basic field. For example:

```
<field name="vmeogroup" type="collection" mime-type="text/plain"
       src="escenic/proxy/vmeo/list/groups?provider=provider-name" select="title">
  <ui:label/>Transcoding Group</ui:label/>
  <group xmlns="http://xmlns.escenic.com/2013/media" enabled="true"/>
</field>
```

A collection field displays a "search as you type" field that lets the user select an entry from an Atom feed - in this case, a list of all VME Online groups owned by a specified VME Online **provider**. For a detailed description of how collection fields work, see http://documentation.vizrt.com/ece-advanced-temp-dev-guide/5.6/collection_fields.html. For information about what a VME Online provider is, see the **Viz Media Engine Administrator's Guide**.

The field element's **src** attribute specifies the URL of the Atom feed that is to be read, and the **select** attribute specifies the name of the Atom feed field that is to be retrieved. Note that the URL in the **src** field is not in fact an absolute URL pointing to a VME Online server, but a relative, local URL. Instead it points to a Content Engine proxy service, which is the recommended way of accessing external Atom feeds. For information on how to set up this proxy service, see **section 3.4**.

Note that it is not possible to change the group of a video or audio clip once it has been created in VME Online (that is, once the content item's

status has been changed from **Draft** to **Submitted**, **Approved** or **Published**). For information about the relationship between content item status and VME Online states, see **section 4.2.3**.

3.2.6 Including Key Frames in Video Content Items

While VME Online is ingesting and transcoding a video object, it can automatically select certain frames and save them as images called **key frames**. These key frame images can be automatically saved as content items in the Content Engine and included as relations of the video content item from which they are derived. In order to achieve this you need to add the following items to your **`content-type`** resource:

A content type for holding the key frames

This content type needs to contain a **`link`** field for storing the key frame images. A minimal key frame content type might look like this:

```
<content-type name="keyframe">
  <ui:label>Key Frame</ui:label>
  <ui:title-field>name</ui:title-field>
  <panel name="main">
    <ui:label>Image content</ui:label>
    <field name="name" type="basic" mime-type="text/plain">
      <ui:label>Name</ui:label>
      <constraints>
        <required>true</required>
      </constraints>
    </field>
    <field type="link" name="binary">
      <relation>com.escenic.edit-media</relation>
      <constraints>
        <mime-type>image/jpeg</mime-type>
        <mime-type>image/png</mime-type>
      </constraints>
    </field>
  </panel>
</content-type>
```

A relation definition

This relation definition is needed to associate key frame content items with their source video content items. It might look like this:

```
<relation-type-group name="default-relation-type-group">
  <relation-type name="keyframes">
    <ui:label>Key frame images</ui:label>
  </relation-type>
</relation-type-group>
```

Relation references

You will need to add a reference to the above relation definition to each of the video content types that you want to include key frames. For example:

```
<content-type name="video">
  <parameter name="com.vizrt.video" value="true"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>External video</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="video" type="basic" mime-type="application/json">
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
    </field>
```

```
    <ref-relation-type-group name="default-relation-type-group"/>
  </panel>
</content-type>
```

A `store-keyframes` element

This element causes the Video plug-in to actually store the key frames generated by VME Online as content items of the correct type. The element must belong to the namespace **`http://xmlns.escenic.com/2010/video`** and must have the following attributes:

- **`content-type`**, specifying the name of the key frame content type you have defined
- **`relation`**, specifying the name of the key frame relation you have defined
- **`state`(optional)**, specifying the state to be applied to generated key frame content items. If not specified, the default state is **`published`**

You must add such an element to the video field in each of the video content types that you want to include key frames. For example:

```
<content-type name="video">
  <parameter name="com.vizrt.video" value="true"/>
  <ui:icon>graphic</ui:icon>
  <ui:label>External video</ui:label>
  <ui:title-field>title</ui:title-field>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="title" type="basic" mime-type="text/plain"/>
    <field name="video" type="basic" mime-type="application/json">
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
      <store-keyframes xmlns="http://xmlns.escenic.com/2010/video"
                       content-type="keyframe" relation="keyframes" state="draft"/>
    </field>
    <ref-relation-type-group name="default-relation-type-group"/>
  </panel>
</content-type>
```

3.3 Configure Viz Content Pilot

The Viz Content Pilot system's **`ax_enableMediaSendToRundown`** parameter must be set to **`y`** in order for the VCP panel browse functionality to work correctly. For details, see the the **Viz Content Pilot User's Guide**.

3.4 Define a Proxy Service for VME Online

In order for the group selection field described in **section 3.2.5** to work you have to define a Content Engine proxy service for VME Online. A proxy service provides a URL inside the Content Engine domain that forwards requests to an external URL (in this case, VME Online). The example group selection field sends Atom feed requests like this:

`escenic/proxy/vmeo/list/groups?provider=`*provider-name*

In order for these requests to work you need to define a proxy service called **`vmeo`** that forwards requests to the URL **`http://`***vme-online-host*`:`*port*`/rest`.

The requests will then be forwarded to **`http://`***vme-online-host*`:`*port*`/rest/list/groups?provider=`***provider-name***.

To create such a proxy service you must:

1. Add a configuration file called *configuration-root*`/com/escenic/webservice/proxy/ProxyResourceConfig.properties` to one of your configuration layers (if you don't already have one).
2. Add the following setting to the file:

```
serviceMapping.vmeo=http://vme-online-host:port/rest
```

For detailed information about Content Engine proxy services and how to set them up, see http://documentation.vizrt.internal/ece-advanced-temp-dev-guide/5.6/content_engine_proxy_service.html.

3.5 Configure VME Production

In order for the VME-based workflows described in **section 1.1.1.1** and **section 1.1.1.2** to work, you need to:

- Either:
  - Ensure that the same user name/password combinations can be used to log in to both Content Studio and Viz Media Studio, or
  - Configure the VME Production system in which your media are archived so that it does **not** require authentication. For a description of how to do this, see "Disabling Authentication" in the **Viz Media Engine Administrator's Guide**.
- Define a Content Engine **named service** for the VME Production system (see **section 3.5.1**).
- Define a Content Engine **proxy authentication service** for the VME Production system (see **section 3.5.2**).
- Enable cross-origin resource sharing on the VME Production system (see **section 3.5.3**).

3.5.1 Define a Named Service for VME

A **named service** is an alias maintained by the Content Engine that references some internal or external URL. The Video plug-in uses the service to ensure that the media resource URLs stored in video and audio content items are insulated from any changes in the URL of the VME production system.

All media URLs stored in video and audio content contain the VME service name rather than the VME system's URL and are translated by the Content Engine when media items are requested. This means that if the URL of the VME Production changes for some reason, then it is only necessary to change the named service definition rather than updating all video and audio content items.

To define the VME named service:

1. Create a file called *configuration-root*`/com/escenic/resolver/NamedServiceResolver.properties` in one of your configuration layers (if it does not already exist).
2. Add the following definition to the file:

   ```
   service.vmep-service=http://vmep-host:port/thirdparty
   ```

   where:

   - *vmep-service* is the name you want to use for the service
   - *vmep-host* is the domain name or IP address of the VME Production system
   - *port* is the port number on which the VME Production system is listening

   The service you define must have the name **`vmeserviceuri`**, and it must point to the VME Production system's **`thirdparty`** service.

3.5.2 Define a Proxy Authentication Service for VME

To add a proxy authentication service for the the VME Production system:

1. Copy **`/opt/escenic/engine/plugins/video/config/studio/com/escenic/studio/service/AuthenticationService.properties`** to your **Content Studio configuration layer** (see note below). For example:

   ```
   $ cp /opt/escenic/engine/plugins/video/config/studio/com/escenic/studio/service/
   AuthenticationService.properties \
        studio-configuration-root/com/escenic/studio/service/
   ```

2. Open the file for editing.
3. Uncomment the service definition in the file and edit it as follows:

   ```
   service.vmep-service=vmep-host:port/thirdparty/
   ```

   where:

   - *vmep-service* is the name you want to use for the service
   - *vmep-host* is the domain name or IP address of the VME Production system
   - *port* is the port number on which the VME Production system is listening

---

The path of the *studio-configuration-root* folder is **`/etc/escenic/engine/studio`** by default, but it can be redefined in the *configuration-layer-root*`/com/escenic/webstart/StudioConfig.properties` configuration file. For further information see Content Studio Configuration Layer.

3.5.3 Enable CORS on the VME Server

Cross-origin resource sharing (CORS) requires the resource-sharing server (VME Production in this case) to include the following headers in its responses:

```
Access-Control-Allow-Methods: GET, OPTIONS
Access-Control-Allow-Origin: http://pub-server:8080
```

where *pub-server* is the name of the publishing server (the Escenic Content Engine in this case).

The VME Production's **`thirdparty`** web service is hosted on Apache, so to add such headers:

1. Open the Apache configuration file **`/var/ardendo/conf/ardome/httpd.conf`** for editing.
2. Add the following entries:

```
LoadModule headers_module  /usr/lib64/apache2-prefork/mod_header
s.so
...
# Enable CORS requests
Header set Access-Control-Allow-Credentials "true"
Header merge Access-Control-Allow-Methods "GET, OPTIONS"
Header merge Access-Control-Allow-Origin "http://pub-server:8080"
```

3. Save your changes.

3.6 Advanced Configuration for VME Online

This section describes some additional configurations you can add in order to improve the integration with VME Online.

3.6.1 Improved Transcoding Control

You can achieve more detailed control over the transcoding process by making use of the information stored in an audio or video item's VDF payload document. In order to enable this functionality:

1. Copy **`/opt/escenic/engine/plugins/video/config/engine/com/escenic/video/adactus/VMEOnlineIngestXMLGenerator.properties`** to your common configuration layer. For example:

```
$ mkdir -p /etc/escenic/engine/com/escenic/video/adactus/
$ cp  /opt/escenic/engine/plugins/video/config/engine/com/escenic/video/adactus/
VMEOnlineIngestXMLGenerator.properties  \
     /etc/escenic/engine/com/escenic/video/adactus/VMEOnlineIngestXMLGenerator.properties
```

2. Open your configuration file for editing.
3. Add the following property:

```
addContentEntry=true
```

Media files are submitted to VME Online for transcoding by sending an XML **input file** that contains the URL of the media file to be transcoded plus information about the required output formats. If you set this **`addContentEntry`** property to **`true`**, then the content item's VDF payload document in is included in the XML input file that is submitted to VME Online.

This option is most commonly used together with an XSL transformation specified with the **`adactusXslFile`** property in the **`TranscodingConfiguration.properties`** file (see **section 3.1.1**). The transformation you specify with the **`adactusXslFile`** property is used to

modify the XML input file before it is sent to VME Online. The idea is that this transformation can use information it finds in the embedded VDF payload to modify other parts of the XML file that affect the transcoding process.

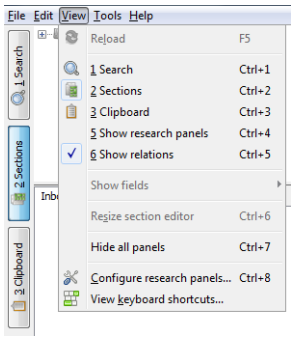
4 Using the Content Studio Plug-in

This chapter mostly describes how to work with video content items in Content Studio on a Windows computer where Vizrt Content Pilot is installed. It is possible to create and open video content items on non-Windows computers or on Windows computers where Vizrt Content Pilot is not installed, but with reduced functionality.

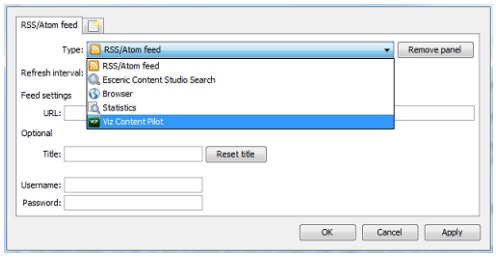
4.1 Adding the Viz Content Pilot Panel

In order to be able to select videos from Viz Media Engine from within Content Studio, you need to enable a special research panel called **Viz Content Pilot**. To enable this panel:

1. Start Content Studio.
2. Select **View** > **Configure research panels...** from the Content Studio menu.

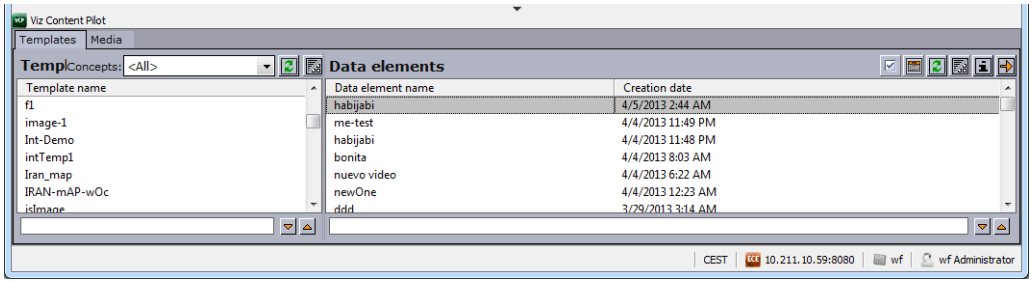


3. Click on the tab in the displayed dialog.
4. Select **Viz Content Pilot** from the **Type** drop-down field.



5. Click **OK**.
6. The **Viz Content Pilot** panel is then added to the research panels area at the bottom of the Content Studio window.

7. To display the **Viz Content Pilot** panel, select **View** > **Show research panels** from the Content Studio menu. You should see something like this:



The **Viz Content Pilot** panel contains the Viz Content Pilot **Newsroom Component**. You can use it to:

- Browse video content stored in a Viz Media Engine.
- Select the video clip you want to add to your content item.
- Decorate the selected video clip with template-based graphics.

For basic instructions on how to do these things, see **section 4.2**.
For more detailed information about the Newsroom Component, see http://documentation.vizrt.com/viz-content-pilot-guide/5.6/newsroom_newsroom_integration.html.

You can only add the Viz Content Pilot panel on Windows PCs where the Viz Content Pilot Newsroom Component is installed. If you are using Content Studio on a different platform or you do not have the Newsroom Component running on your PC, you can still browse VME video content using Viz Media Studio, and add video clips to content items by drag-and-drop. You cannot, however, add template-based graphics to video clips.

4.2 Working With Media Content Items

This section describes how you can use Content Studio to:

- Create video and audio content items
- Use the video and audio timeline editors to:
  - Crop video/audio items
  - Add cue points to video/audio items
  - Decorate video/audio items with transient links to related content
- Publish video and audio content items

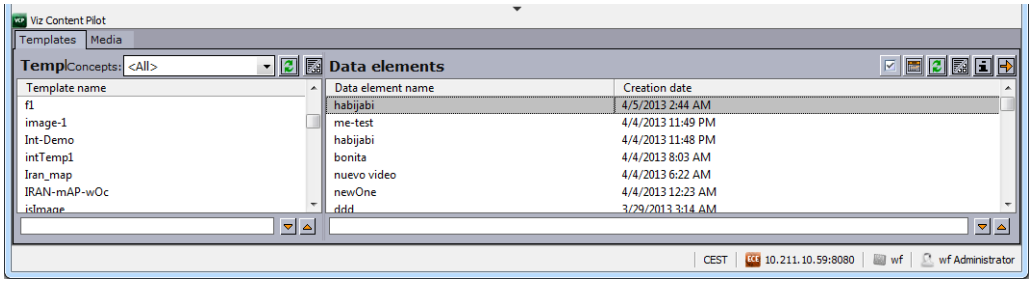
4.2.1 Creating a Video Content Item

How you create a video content item depends upon whether you are creating an external video content item (where the video clip is stored in Viz Media Engine) or an internal video content item (where you upload a video from your local machine to be stored in the Content Engine).

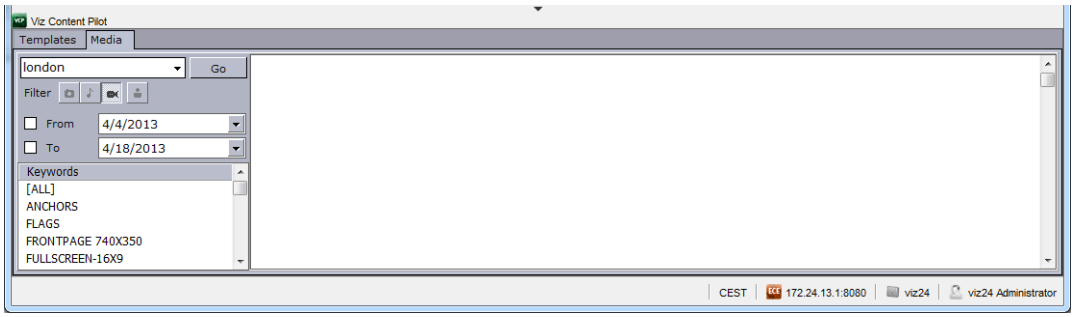
4.2.1.1 Creating an External Video Content Item (VCP Method)

You can only use this method if you have added the **Viz Content Pilot** panel to Content Studio (see **section 4.1**).

1. If necessary, select **View** > **Show research panels** from the Content Studio menu to display the **Viz Content Pilot** panel.

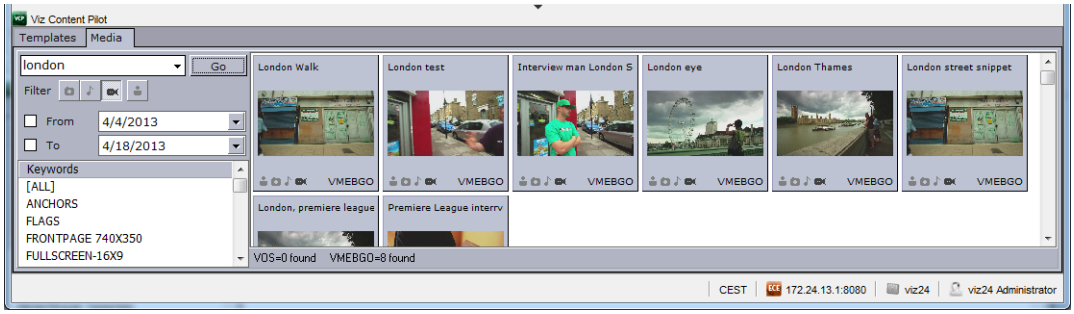


2. Click on the **Media** tab.
3. Specify search criteria using the controls on the left side of the panel:

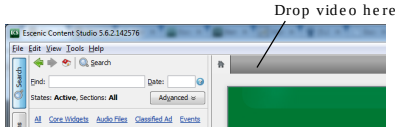


For detailed information about how to use the **Media** tab this, see http://documentation.vizrt.com/viz-content-pilot-guide/5.6/newsroom_newsroom_integration.html.

4. Click on **Go**. Thumbnails of matching videos are then displayed on the right side of the panel:



5. Pick a video thumbnail and drag it into the Content Studio work area. You must drop it right at the top of the work area where content editor tabs are displayed:



When you drop the video, a new content item of a suitable type is created, and the video is added to it.

6. Enter the required values into the content item's fields (you will usually need to add at least a title) and click **Save**.

Alternatively, you can create the content item first (by selecting **File** > **New** > **<your-external-video-content-type>** from the Content Studio menu in the usual way), and then select a video and add it to the content item you have created. If you do this, however, then you have to make sure that you drop the video you select inside the marked area of the content item's video field (a dotted box). If you drop it anywhere else in the content editor, then instead of adding the video to your content item, Content Studio will either do nothing or create a new content item for it.

4.2.1.1.1 Adding Graphics to an External Video

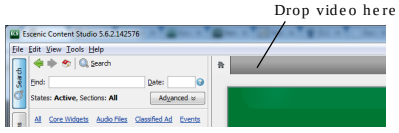
If required, you can add template-based graphics to your selected video before adding it to a content item. In order to do this:

1. Right-click on the video thumbnail and select **Open in Timeline Editor** from the displayed menu. The video is then displayed in the timeline editor.
2. Click on the **Add** button. This displays a new page that you can use to select and fill out a graphic template. (For detailed information about how to do this, see http://documentation.vizrt.com/viz-content-pilot-guide/5.6/newsroom_newsroom_integration.html).
3. When you have filled out your selected template, click on **OK**. The video is displayed again with the template graphics added. You can play the video again and adjust the timing of the graphic by dragging in the time line displayed below the video play controls.
4. You can add further template-based graphics by clicking on **Add** again. When you are satisfied with the video, click **OK** to select it.

4.2.1.2 Creating an External Video Content Item (VMS Method)

You can use this method to add videos from a VME Video Production system on any platform. Viz Content Pilot does not need to be installed. You do, however, need access to Viz Media Studio. Viz Media Studio is a browser-based application that does not need to be installed on your PC - you just need the correct URL and valid log-in credentials.

1. Use Viz Media Studio to find the video clip you want to use. For details of how to use Viz Media Studio, see the **Viz Media Studio User's Guide**.
2. Drag your selected video clip into the Content Studio work area. You must drop it right at the top of the work area where content editor tabs are displayed:



When you drop the video, a new content item of a suitable type is created, and the video is added to it.

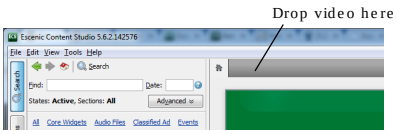
3. Enter the required values into the content item's fields (you will usually need to add at least a title) and click **Save**.

Alternatively, you can create the content item first (by selecting **File** > **New** > **<your-external-video-content-type>** from the Content Studio menu in the usual way), and then select a video in Viz Media Studio and add it to the content item you have created. If you do this, however, then you have to make sure that you drop the video you select inside the marked area of the content item's video field (a dotted box). If you drop it anywhere else in the content editor, then instead of adding the video to your content item, Content Studio will either do nothing or create a new content item for it.

4.2.1.3 Creating an Internal Video Content Item

To create an internal video content item:

1. Use Explorer (Windows) or Finder (Mac) to find the video clip you want to use.
2. Drag your selected video clip into the Content Studio work area. You must drop it right at the top of the work area where content editor tabs are displayed:



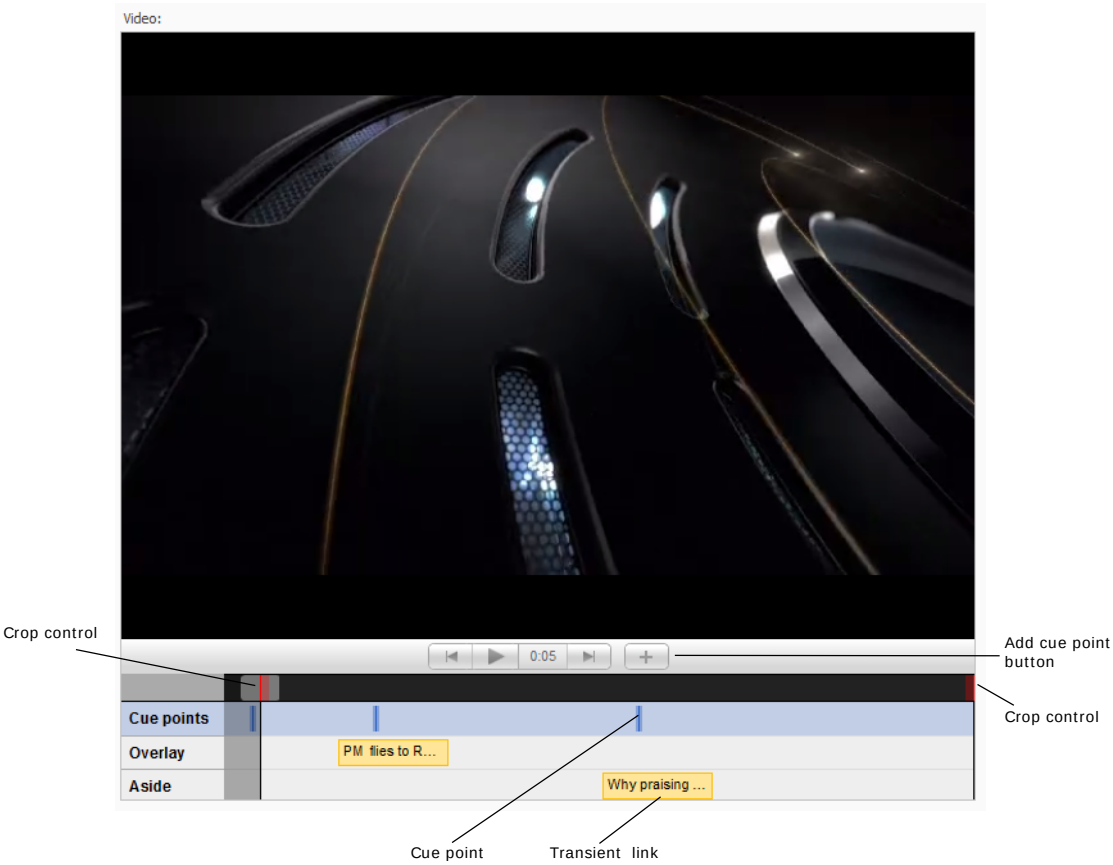
When you drop the video, a new content item of a suitable type is created, and the video is added to it.

3. Enter the required values into the content item's fields (you will usually need to add at least a title) and click **Save**.

Alternatively, you can create the content item first (by selecting **File** > **New** > **<your-internal-video-content-type>** from the Content Studio menu in the usual way). A **File Open** dialog is then displayed, allowing you to select the video file you want to upload.

4.2.1.4 Using the Video Timeline Editor

The video in a video content item is displayed in a **timeline editor**:



The timeline editor is a video player that incorporates a number of simple editing functions for:

Cropping
That is, removing footage from the start and/or end of the video.

Adding cue points
A cue point marks a point of interest in the video (the start of a particular scene or event). It can be used to make a link that starts playback at the cue point.

Adding transient links
A transient link is a link that appears at a certain point during video playback and/or disappears at a certain point. It makes it possible to display links that are relevant to the content being viewed by users.

All of these functions involve placing components on the **timeline**. The timeline is the thick black stripe displayed below the video playback controls, which represents the playing time of the video. When the video is played back, a cursor representing the current frame moves along the timeline. You can move the cursor to specific frames by clicking in the timeline.

To crop the video

You can crop the video by simply dragging the red crop controls at either end of the timeline towards the center of the timeline. You can move them back and forth as much as you want and play back the video to check the results. Note that cropping in this way does not modify the original video in any way, it just lets you select the segment you want to publish.

To add a cue point

Position the cursor at the required point by clicking in the timeline. Then click on the **+** button alongside the video playback controls. An **Edit Cue Point** dialog is then displayed in which you can enter a **Title** and a **Description** for the cue point, and then click on **OK** to create the cue point. You can move existing cue points by dragging them along the time line. If you hold the mouse pointer over a cue point, (edit) and (delete) buttons are displayed. Clicking on redisplays the **Edit Cue Point** dialog, and clicking on deletes the cue point.

Exactly how the title and description you enter are used depends upon the templates in your publication. Typically, the title is used as link text and the description as some kind of tooltip/hover text.

To add transient links

Drag the content item to which you want to create a link into one of the transient link fields and drop it approximately where you would like it to appear in the timeline. An **Edit Timeline Element** dialog is displayed in which you can enter a **Title**, and **Start time**/**End time** values and then click on **OK** to create the link.

You can also adjust the link's start and end times by dragging the link along the time line, and dragging its start and end points. If you hold the mouse pointer over a link, (edit) and (delete) buttons are displayed. Clicking on redisplays the **Edit Timeline Element** dialog, and clicking on deletes the link.

By default, the timeline editor has two transient link fields called **Overlay** and **Aside**. The **Overlay** field is intended to hold transient links that appear as overlays (that is, inside the video frame) and the **Aside** field is intended to hold transient links that appear somewhere nearby but outside the video frame. Exactly how the links are displayed in publications, however, is determined by the template developer.

The transient link fields may vary between installations. They are not necessarily called **Overlay** and **Aside**, and there may be more (or less) than two of them.

4.2.2 Creating an Audio Content Item

You can create an audio content item in all the same ways that you can create a video content item. The only differences are:

- When adding external audio via Viz Content Pilot (see **section 4.2.1.1**), you cannot add graphics to the audio, as you can for a video.
- The transient link field name **Overlay** has no particular meaning in the case of audio content items.

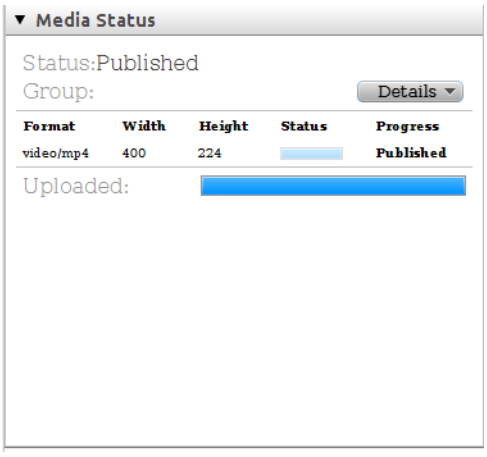
4.2.3 Publishing Media Content Items

The general process of publishing a content item in Content Studio involves pushing it through a series of states. When you first save a new content item, it is saved in the state **draft**. It can then be pushed through the states **submitted** (that is, submitted for approval), **approved** and **published**. A content item is only visible to users of your web site when it reaches the **published** state. Content Studio does not enforce the use of all these states - it is perfectly possible to skip **submitted** and **approved**, and move a content item straight from **draft** to **published** if you have sufficient access rights.

Publishing media content items is a rather more complex process than publishing text and image content items, because it is not just an administrative process - it also involves:

- Transcoding the video/audio content to a variety of formats and encodings for display/playback on different devices and platforms
- Copying the transcoded videos/audios to their published destinations (often an external content delivery network)

These processes are carried out by VME Online, and VME Online has its own set of states used to represent various stages in this process. Moving a content item from one state to the next in Content Studio actually initiates a VME Online process, and can cause the video/audio content item to move through a sequence of several VME Online states, which are reported back to Content Engine and displayed in a special **Media Status** section of the Content Studio attributes panel:



This section is only present in video/audio content item editors. It shows the current VME Online status of the video/audio, current transcoding group name and other progress-related information. There is a **Compact**/**Details** button that you can use to control the amount of information displayed.

The following table shows the sequence of **Media Status** messages corresponding to each Content Studio state, and describes the VME Online states the messages represent

Content Studio state	**Media Status/VME Online state**
Draft	none/none: no information about the video/audio has been sent to VME Online
Submitted	**Started**/none: the VME Online transcoding process has not yet been started **Downloading**/**Downloading** **Transcoding**/**Processing** **Transcoded**/**Ready**
Approved	**Publishing**/none: VME Online has been asked to publish the transcoded videos/audios but has not completed the operation and returned a new status **Published**/**Published**
Published	**Published**/**Published**

As with other content types, you can skip states and publish a **draft** content item without first moving it through the intermediate states (if you have sufficient access rights). If you do so, however, you will see that the content item in fact does pass through the intermediate states as the video/audio clip is processed by VME Online.

For video/audio content items there is no distinction between the states **approved** and **published**. If you move a content item to the approved state then it ends up **published**.

Content Studio state changes only trigger VME Online processes and state changes when you move the content item forward through the states. No VME Online activity is triggered if you unpublish a **published** content item by changing its state back to **draft**.

Two other **Media Status** messages you may see are:

Failed
: This represents the VME Online status **failed**. It indicates that transcoding has failed.

Restarted
: This means that VME Online has been asked to delete the previously transcoded video/audio clips and is transcoding a new clip.

5 Automated Media Publishing

This section contains detailed information about how to implement the automated publishing workflows described in **section 1.1.2**. Whichever method you use, the basic objective is the same. You need to:

1. Define a content item of the correct type in the Content Engine. In this case it must be an external video content type as described in **section 3.2.1** or an external audio content type as described in **section 3.2.3**. The definition must include all required fields. The most important of these is the video field described in **section 3.2.1** or the audio field described in **section 3.2.3**, which must contain JSON data specifying the URI and MIME type of the video/audio to be published, plus optional additional information specifying crop points, cue points and transient links.
2. Create the defined content item.
3. Change the status of the content item to **`published`**. This causes the Content Engine to send the video/audio to VME Online for transcoding and key frame generation.

5.1 Web Service-Based Automation

The recommended way to implement an automated video publishing process is to use the Content Engine's web service. This web service allows you to interact with the Content Engine by sending HTTP requests to the web service. For a description of the web service and how to use it in general, see the Escenic Content Engine Integration Guide. The web service allows an external process to perform most of the operations that end users can perform from Content Studio. In this case the objective is to create a video/audio content item, which involves the following steps:

1. Get the web service URI for the publication/section to which the content item is to be added (see http://documentation.vizrt.com/ece-integration-guide/5.6/navigate_the_section_hierarchy.html).
2. Make an XML document containing all the information required to create the required content item and upload it to the correct URI. This is described in general terms here: http://documentation.vizrt.com/ece-integration-guide/5.6/add_a_content_item.html.

When you are creating a video/audio content item, the content of the VDF payload described in http://documentation.vizrt.com/ece-integration-guide/5.6/add_a_content_item.html must:

- Have a structure that matches the video/audio content type for your publication
- Contain correctly structured JSON data in the content item's video/audio field that specifies the URI and MIME type of the video/audio to be published

The following example shows a VDF payload for creating a video content item, based on the content type example given in **section 3.2.1**:

```
<vdf:payload xmlns:vdf="http://www.vizrt.com/types" model="http://my-content-engine/webservice/
publication/my-publication/escenic/model/external-video">
  <vdf:field name="title">
    <vdf:value>My First Video</vdf:value>
  </vdf:field>
  <vdf:field name="video">
    <vdf:value>
      { "master" : { "uri" : "http://my-video-server/video1.mpeg", "mime-type" : "video/mpeg"}}
    </vdf:value>
  </vdf:field>
</vdf:payload>
```

Here is another example which shows a VDF payload for creating an audio content item, based on the content type example given in **section 3.2.3**:

```
<vdf:payload xmlns:vdf="http://www.vizrt.com/types" model="http://my-content-engine/webservice/
publication/my-publication/escenic/model/external-audio">
  <vdf:field name="title">
    <vdf:value>My First Audio</vdf:value>
  </vdf:field>
  <vdf:field name="audio">
    <vdf:value>
      { "master" : { "uri" : "http://my-audio-server/audio1.mp3", "mime-type" : "audio/mp3"}}
    </vdf:value>
  </vdf:field>
</vdf:payload>
```

5.2 Syndication-Based Automation

You can also implement an automated video/audio publishing process using the Content Engine's syndication format. This is a proprietary Vizrt XML file format that can be used for import/export purposes. It is described in detail in the Escenic Content Engine Syndication Reference.

To define and create a content item in this way you simply create a syndication file containing all the required information and upload it to a specified import folder on the Content Engine server. The Content Engine will then automatically import the file and create a corresponding content item, triggering the transcoding and key frame generation process.

For a general description of how to use the Content Engine's import service, see http://documentation.vizrt.com/ece-syndication-ref/5.6/the_import_service.html.

The following example shows a syndication file that could be used to import a video content item, based on the content type example given in **section 3.2.1**:

```
<?xml version="1.0" encoding="UTF-8"?>
<escenic xmlns="http://xmlns.escenic.com/2009/import" version="2.0">
  <content source="my-video-library" sourceid="1" type="external-video" state="published">
    <section-ref unique-name="videos" home-section="true"/>
    <field name="title">My First Imported Video</field>
    <field name="video">
      { "master" : { "uri" : "http://my-video-server/video1.mpeg", "mime-type" : "video/mpeg"}}
    </field>
  </content>
</escenic>
```

Following is another example which shows a syndication file that could be used to import an audio content item, based on the content type example given in **section 3.2.3**:

```
<?xml version="1.0" encoding="UTF-8"?>
<escenic xmlns="http://xmlns.escenic.com/2009/import" version="2.0">
  <content source="my-audio-library" sourceid="1" type="external-audio" state="published">
    <section-ref unique-name="audios" home-section="true"/>
    <field name="title">My First Imported Audio</field>
    <field name="audio">
      { "master" : { "uri" : "http://my-audio-server/audio1.mp3", "mime-type" : "audio/mp3"}}
    </field>
  </content>
</escenic>
```

The video/audio fields (highlighted in the examples above) must contain the URI and the MIME type of the video or audio clip, and must be specified in JSON format, since video/audio fields are defined with a MIME type of **`application/json`** (see **section 3.2.1** and **section 3.2.3**).

Note also that, in order to trigger transcoding, the content items must be published on import, by setting the **`content`** element's **`state`** attribute to **`published`**.

Although it is technically possible to import internal videos/audios via syndication files, this is not described here. Local upload of videos/audios to the Content Engine is primarily intended for small scale, casual use and not considered suitable for automated workflows (see **section 1.1.1.3**).

5.3 Working With the Video/Audio Field

This section describes how you can use the video/audio field to import different types of video/audio. The video/audio field must contain the URI and the MIME type of the video/audio specified in JSON format. You can, however, also include a lot of other information about the imported audio or video clip, such as crop points, cue points and transient links. The field must contain a map of one or more key-value pairs, for example:

```
{
  key1:value1
  key2:value2
}
```

The values may themselves also be maps:

```
{
  key1:{
    key1a:value1a
    key1b:value1b
  }
  key2:value2
}
```

and so on. The following sections describe the specific key-value pairs that can appear in a video or audio field, and what they mean.

5.3.1 Specifying Basic Data

You can specify basic information (URL, MIME type and so on) about a video or audio clip in one of two ways, depending on where the video or audio clips are being loaded from.

5.3.1.1 Loading From Other External Sources

If you are loading the video and audio clips from an external source other than a VME online system, then instead of **`source`** you must include a key-value pair called **`master`**. This is a much simpler structure:

```
{
  "master":{
    "uri" : "http://my-video-server/video1.mpeg",
    "mime-type" : "video/mpeg"
  }
  ...
}
```

The value of the **`master`** key must be a map containing the following key-value pairs:

`uri`
The URI of the actual video/audio clip.

`mimeType`
The mime-type of the video/audio clip.

5.3.2 Specifying Crop Points

You can specify that the loaded video or audio clip is to be cropped before it is transcoded. To do this you must include a **`playback`** key-value pair in the map. For example:

```
{
  ...
  "playback":{
     "in":1.5,
     "out":5.6
  }
}
```

The value of the **`playback`** key must be a map containing the following key-value pairs:

`in`
The point at which the transcoded, published clip is to start, measured from the start in seconds.

`out`
The point at which the transcoded, published clip is to end, measured from the start in seconds.

5.3.3 Specifying Timeline Information

You can also include timeline information (cue points and transient links). To do this you must include a **`timeline`** key-value pair in the map:

```
{
  ...
  "timeline":{
    "cuepoints":cuepoints-value,
    "tracks":tracks-value
  }
}
```

The value of the **`timeline`** key must be a map containing the following key-value pairs:

`cuepoints`
An array of one or more cue point definitions (see **section 5.3.3.1**).

`tracks`
A map containing default transient link definitions for the transient link tracks defined at your installation (see **section 5.3.3.2**). By default there are two such tracks, called **Overlay** and **Aside**.

5.3.3.1 Specifying Cue Points

Cue points are defined in an array. Each value in the array is a map of key-value pairs:

```
{
  ...
  "timeline":{
    "cuepoints":[
      {
        "id":"cue1",
        "time":"3.176",
        "title":"cue one",
        "description":"this is cue one"
      },
      {
        "id":"cue2",
        "time":"7.520",
        "title":"cue two",
        "description":"this is cue two"
      },
      ...
    ],
    "tracks":tracks-value
  }
}
```

The key-value pairs are:

`id`
An ID for the cue point.

`time`
The point at which the cue point is to be placed, measured in seconds from the start of the original uncropped clip.

`title`
The title of the cue point.

`description`
The description of the cue point.

5.3.3.2 Specifying Transient Links

The **`timeline`** map can contain one key-value pair for each defined transient link track. If your installation has the standard two tracks called **Overlay** and **Aside**, then you can add transient links by including the keys **`OverlayDefault`** and **`AsideDefault`** in the **`timeline`** map. The actual links to be included in each track are defined in an array, and each value in the array is a map of key-value pairs:

```
{
  ...
  "timeline":{
    "cuepoints":tracks-value,
    "tracks":{
      "OverlayDefault":[
        {
          "contentID": "117",
          "startTime": 20.0609756097561,
          "endTime": 23.262195121951216,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        },
        {
          "contentID": "118",
          "startTime": 33.87195121951219,
          "endTime": 39.96951219512194,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        }
      ],
      "AsideDefault":[
        {
          "contentID": "114",
          "startTime": 5.48780487804878,
          "endTime": 8.536585365853657,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        }
      ]
    }
  }
}
```

The key-value pairs are:

`id`
: An ID for the linked conent.

`startTime`
: The point at which display of the link is to start, measured in seconds from the start of the original uncropped clip.

`endTime`
: The point at which display of the link is to end, measured in seconds from the start of the original uncropped clip.

`title`
: The title of the link.

5.3.4 VME Online Status URI

Once a video or audio content item is submitted for transcoding by VME Online, VME online returns the URI to which it publishes transcoding status information. The Content Engine adds this URI to the JSON map in the content item's video/audio field. For example:

```
{
```

```
  ...
  "status-uri":"http://vme-online-host/rest/digitalItem/status/146"
}
```

Your application can therefore retrieve status information about transcoding progress by sending requests to the content item's **`status-uri`**.

It is possible to define workflows where video or audio clips are submitted to VME Online for transcoding before they are added to content items, in which case you would need to add the **`status-uri`** key-value pair yourself. No such workflows are, however, described in this manual.

---

6 Accessing Media from Templates

You can access the transcoded versions of a video/audio clip from your JSP templates using JSTL as follows (in all the examples *video-field-name* is the name of the video field in your content type and *audio-field-name* is the name of the audio field in your content type and *index* is the index of the transcoded version you want):

- To access the URI of a transcoded video stored on the VME Online server:

  ```
  ${article.fields.video-field-name.value.video[index].uri}
  ```

- To access the video's MIME type:

  ```
  ${article.fields.video-field-name.value.video[index]['mime-type']}
  ```

- To access the video's height:

  ```
  ${article.fields.video-field-name.value.video[index].height}
  ```

- To access the video's width:

  ```
  ${article.fields.video-field-name.value.video[index].width}
  ```

- To access the video's duration:

  ```
  ${article.fields.video-field-name.value.duration}
  ```

- To access the video's timeline information:

  ```
  ${article.fields.video-field-name.value.timeline.cuepoints}
  ```

  ```
  ${article.fields.video-field-name.value.timeline.tracks}
  ```

  ```
  ${article.fields.video-field-name.value.playback}
  ```

- To access the URI of a transcoded audio file stored on the VME Online server:

  ```
  ${article.fields.audio-field-name.value.audio[index].uri}
  ```

- To access the audio file's MIME type:

  ```
  ${article.fields.audio-field-name.value.audio[index]['mime-type']}
  ```