Video

Plug-in User Guide

3.0.2.164607

escenic
A CCI COMPANY

Table of Contents

1 Introduction

The Escenic Content Engine's Video plug-in extends Content Studio with the functionality required to efficiently manage web site media content.

The Video plug-in provides:

- Fully-automated transcoding of video and audio content using Amazon Elastic Transcoder
- Fully-automated distribution of published video and audio content using Amazon CloudFront
- Simple Video and Audio editing functionality in Content Studio
- Optional integration with external media storage solutions such as media asset management systems

> Audio content is not supported in the current version.

The Video plug-in's objective is to make handling media content in the context of web site production as painless as possible, so the editing functionality added to Content Studio is a very simple **timeline editor** that is specifically geared to the needs of online publishing. The timeline editor allows you to:

- **Crop** an audio or video clip (that is select the segment you actually want to publish)
- Add **cue points** (links that allow viewers/listeners to go directly to points of interest in the clip)
- Add **transient links** to a clip (links to related content that appear and disappear while a clip is playing)
- Select a **poster frame** for a video clip (a still that can be used to represent the clip when it is not playing)

All of these editing functions are **non-destructive**: that is, they do not modify the source media object, they only affect the published result.

The source media objects can either be **internal** or **external**, according to how they are stored:

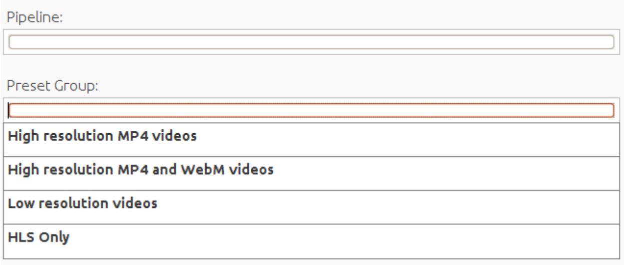
- An **internal** media object is one that is stored under the control of the Content Engine, in exactly the same way as an image or any other kind of binary file. A media object of this kind is usually acquired by uploading to Content Studio (although they can also be imported in various ways, just like any other content items).
- An **external** media object is one that is stored somewhere on the net, most likely in a media asset management (MAM) system such as Vizrt's VizOne, or in some other storage solution. A media object of this kind is usually supplied to the Video plug-in via an integration interface which may be:
  - The Content Engine web service
  - The Content Engine syndication subsystem
  - Drag and drop between the source application and Content Studio

  > External media objects are not supported in the current version.

> In order for the Video plug-in to work, the Content Engine must use Amazon S3 as its back-end storage for binary files. For more about this, see chapter 2.

1.1 About Amazon Elastic Transcoder

Amazon Elastic Transcoder is an Amazon web service that provides on-demand transcoding of media files between all commonly used audio and video formats. Once the Video plug-in has been correctly configured, it is largely invisible to plug-in users, simply providing a background service. Two Elastic Transcoder concepts will, however, often be visible: when publishing a media file, the user will usually be able to choose between different **pipelines** and **preset groups**:


Pipeline:
Preset Group:
High resolution MP4 videos
High resolution MP4 and WebM videos
Low resolution videos
HLS Only

A **pipeline** is simply an Elastic Transcoder job queue. In a typical set-up there are two pipelines, one for normal jobs and one for high priority jobs, and the Content Studio user can choose which to use when publishing a media content item. There is no actual difference between the two pipelines other than their name, so the high priority pipeline will only work as a "fast lane" if it is actually reserved for a small number of high priority jobs. Pipelines are defined in Elastic Transcoder.

A **preset** is a set of parameters defining in great detail the output required from a transcoding job. For a video transcoding, it defines the file container format, video and audio codecs, bit rate, frame rate, width and height and so on. Presets are defined in Elastic Transcoder. A **preset group** is a named group of presets defined during the configuration of the Video plug-in. The idea is that a preset group should contain all the presets required to publish a particular type of media file in a particular context. Typically, a user will only need to choose between a small number of preset groups ("Standard Video" and "High Res Video", for example) and in some cases may not need to choose at all, since only one preset group has been defined.

The pipelines and preset groups available to the Video plug-in are defined in XML files called **transcoder configuration files**. Several of these files may be defined in order to provide different set-ups for different publications. For further information, see section 3.4.

2 Installation

The following preconditions must be met before you can install Video 3.0.2.164607 in a Content Engine:

- Escenic Content Engine version 5.7.37.163612 and Escenic assembly tool have been installed as described in the Escenic Content Engine Installation Guide and are in working order.
- You have set up an Amazon S3 account that you can use for storing video files. For general information about how to do this, and how to configure the Content Engine to use S3 for storage of binary files, see Amazon S3 Storage. More specific instructions about the Video plug-in's use of S3 storage is provided in chapter 3.
- You have set up an Amazon Elastic Transcoder account, and defined the **pipelines** and **presets** you are going to use. For details about how to do this, see the Elastic Transcoder documentation
- You have the required plug-in distribution file `video-dist-3.0.2.164607.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 5.7.37.163612. *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/57/video-
dist-3.0.2.164607.zip
```

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

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

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

4. Unpack the downloaded distribution files:

   ```
   $ cd /opt/escenic/engine/plugins
   $ unzip /mnt/download/video-dist-3.0.2.164607.zip
   $ unzip /mnt/download/vcpeditor-dist-n.n.zip
   ```

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

5. 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.

6. > 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**.

7. 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:

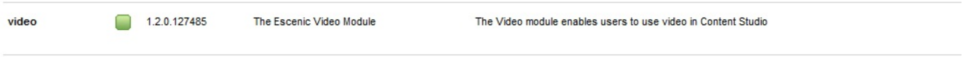
🟩

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:



video	🟩	1.2.0.127485	The Escenic Video Module	The Video module enables users to use video in Content Studio

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 configure it correctly with the values needed to access and use various external services:

- Amazon Elastic Transcoder
- Amazon S3
- Amazon Cloudfront

The configuration settings need to be made in the following places:

- The Content Engine's common configuration layer (see Configuring The Content Engine for general information about Escenic configuration layers).
- Your publications' application configuration layers. An application configuration layer works in more or less the same way as other Content Engine configuration layers, but it is publication-specific.
- One or more **transcoder configuration files**. These are XML files containing details of the Elastic Transcoder **pipelines** and **presets** to be used by your publications.
- Your publication's section parameters
- Your publications' `content-type` resource files (see The Publication Resources for general information about Escenic resource files).

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 Logging.

3.1 Quick Start

You don't necessarily need to carry out **all** the configuration tasks described in this chapter in order to get the Video plug-in running. The following list describes the fastest route to a working media production line.

1. Copy example configuration files to your common configuration layer as described in section 3.2.
2. Edit `/com/escenic/storage/filesystems/VideoFileSystemConfiguration.properties` as described in section 3.2.1.
3. Edit `/com/escenic/storage/filesystems/KeyframeFileSystemConfiguration.properties` as described in section 3.2.2.
4. Edit `/com/escenic/storage/Storage.properties` as described in section 3.2.3.
5. Edit `/etc/escenic/engine/common/com/escenic/media/aws/DefaultTranscoderConfig.properties` as described in section 3.2.4.
6. Edit `/etc/escenic/engine/common/com/escenic/media/services/AWSClientConfig.properties` as described in section 3.2.6.

7. Edit **`/etc/escenic/engine/common/com/escenic/media/aws/transcode-config.xml`** as described in section 3.4.
8. Edit **`/etc/escenic/engine/common/com/escenic/media/aws/preview/PreviewTranscoderConfig.properties`** as described in section 3.2.7.
9. Add media content types to your publication content-type resources as described in section 3.6.1.

This quick-start configuration will give you an installation that:

- Uses the same transcoder configuration for all publications
- Uses S3 for storing transcoded media files, not Amazon CloudFront

If this does not meet all your requirements, then you will need to edit some additional configuration files, as described in the following sections.

3.2 Common Configuration Layer Set-up

This set-up task consists of copying example configuration files from the plug-in installation into the Content Engine common configuration layer and then modifying the copied files to meet your requirements. In detail, you must:

1. Log in to your Content Engine host as the **`escenic`** user.
2. Copy all the supplied common configuration layer files as follows:

```
$ cp -r /opt/escenic/engine/plugins/video/misc/siteconfig/common/com/escenic \
> /etc/escenic/engine/common/com
```

3. Edit the copied files as described in the following sections.
4. Edit **`StudioConfig.properties`** (which will already be in your common configuration layer) as described in section 3.2.10.

3.2.1 VideoFileSystemConfiguration.properties

The full path of this file is **`/com/escenic/storage/filesystems/VideoFileSystemConfiguration.properties`**.This file is used to define the cloud storage location in which you will store your video files.

For a detailed description of how to configure cloud storage file system components in general, see Create FileSystemConfiguration Components.

3.2.2 KeyframeFileSystemConfiguration.properties

The full path of this file is **`/com/escenic/storage/filesystems/KeyframeFileSystemConfiguration.properties`**. It is used to define the cloud storage location in which Amazon Elastic Transcoder stores the key frame files it generates during transcoding.

The key differences between this file and **`/com/escenic/storage/filesystems/VideoFileSystemConfiguration.properties`** are that:

- The **`baseURI`** property must be set to point to the Amazon bucket that you set up as the thumbnail output bucket when configuring Amazon Elastic Transcoder

- It has the following additional property setting:

  ```
  readOnly=true
  ```

  which tells the Content Engine that it may not use this S3 location for storing files, only for reading them.

For a detailed description of how to configure cloud storage file system components in general, see Create FileSystemConfiguration Components.

3.2.3 Storage.properties

The full path of this file is **`/com/escenic/storage/Storage.properties`**. If you don't already have a copy of this file in your common configuration layer, then you will need to create it. It determines which storage systems the Content Engine uses for different kinds of files. It needs to reference the **`VideoFileSystemConfiguration`** and **`KeyframeFileSystemConfiguration`** components you have defined (plus possibly other cloud storage locations that you may have defined previously).

For a detailed description of how to configure this file in general, see Create FileSystemConfiguration Components.

> When you are setting the **`fileSystemConfigurations`** property in this file, you must be sure to use **absolute paths** to specify the location of the **`VideoFileSystemConfiguration`** and **`KeyframeFileSystemConfiguration`** components (and any other file system components that will be used by the Video plug-in, should there be any). For example:
>
> ```
> fileSystemConfigurations=/com/escenic/storage/filesystems/
> KeyframeFileSystemConfiguration,/com/escenic/storage/filesystems/
> VideoFileSystemConfiguration
> ```
>
> This is only required for file system components used by the Video plug-in, it is not a general requirement.

3.2.4 DefaultTranscoderConfig.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/aws/DefaultTranscoderConfig.properties`**. You need to set the following property in this file:

`serviceDefinition`

The path to the XML configuration file containing your transcoding service configuration. For example:

```
serviceDefinition=transcode-config.xml
```

For further information, see section 3.4.

If you want to have different transcoding set-ups for different publications, then you will need to create several copies of this file with different names (for example **`TranscoderConfigPub1.properties`**, **`TranscoderConfigPub2.properties`** and so on, and set the **`serviceDefinition`** property to reference a different transcoder configuration file in each copy. For example:

```
serviceDefinition=pub1-transcode-config.xml
```

For each additional copy of this file that you create, you also need to add an entry referencing the file to **`/etc/escenic/engine/common/com/escenic/media/aws/TranscodingConfigFactory.properties`** (see section 3.2.5).

3.2.5 TranscodingConfigFactory.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/aws/TranscodingConfigFactory.properties`**.

> If you only want one transcoding set-up for all your publications, then you do not need to edit this file.

If you want to have different transcoding set-ups for different publications, then you need to set the following properties:

`defaultTranscodingConfig`

This property must be set to reference your default transcoder configuration, for example:

```
defaultTranscodingConfig=./DefaultTranscoderConfig
```

`publicationTranscodingConfigs`*.pub-name*

If you have made copies of **`DefaultTranscoderConfig.properties`** containing different transcoding configurations for different publications, then you can specify several **`publicationTranscodingConfigs`** properties to reference them, 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 a specialized transcoding configuration, for example:

```
publicationTranscodingConfigs.DailyNews=./TranscoderConfigDailyNews
publicationTranscodingConfigs.WeeklyGossip=./TranscoderConfigEntertainment
publicationTranscodingConfigs.Entertainment=./TranscoderConfigEntertainment
```

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 with the **`defaultTranscodingConfig`** property is used.

3.2.6 AWSClientConfig.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/services/AWSClientConfig.properties`**. You need to set the following properties in this file:

`accessKey`

An access key for accessing your Amazon Web Services (AWS) account. This is the access key you got when you set up your Amazon S3 storage (see Amazon S3 Storage). For information about AWS access keys and AWS identity management in general, see the AWS IAM documentation.

```
accessKey=aws-access-key
```

`secretKey`

The secret key for the access key specified with the **`accessKey`** property. This is the access key you got when you set up your Amazon S3 storage (see Amazon S3 Storage).

```
secretKey=aws-secret-key
```

`region`

The name of the AWS Elastic Transcoder region you want to use. For example:

```
region=US_EAST_1
```

For a list of the currently available Elastic Transcoder regions, see Amazon Elastic Transcoder.

`protocol`

The protocol to use when generating pre-signed URLs for accessing content stored in S3. The allowed values are **`HTTP`** (the default) and **`HTTPS`**. For example:

```
protocol=HTTP
```

3.2.7 PreviewTranscoderConfig.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/aws/preview/PreviewTranscoderConfig.properties`**.

In order for the Content Studio media player / timeline editor to work it requires low resolution versions (often called **proxy versions**) of media objects to work with. As soon as a media object is uploaded or imported into a media content item, therefore, the Video plug-in sends it to Amazon Elastic Transcoder to be transcoded to the required proxy format. **`PreviewTranscoderConfig.properties`** contains the configuration settings required to access the Elastic Transcoder for this purpose.

If you do not define a **`PreviewTranscoderConfig.properties`** file, then the Content Studio media player / timeline editor will not work. You will be able to publish media content, but you will not be able to preview or edit it in Content Studio.

You need to set the following properties in this file:

`pipelineID` (required)

The ID of the Elastic Transcoder pipeline to which proxy version transcoding jobs are to be submitted. For example:

```
pipelineID=pipeline-id
```

`videoPresetID` (required)

The ID of the Elastic Transcoder preset you have created for generating video proxy versions. For example:

```
videoPresetID=preset-id
```

3.2.8 Poller.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/status/Poller.properties`**.

The settings in this file control the service that submits media content for transcoding.

You can **optionally** set the following properties in this file:

`enableService`

Determines whether or not the service is enabled. It is enabled by default on all hosts, although it only ever actually runs on one host. You should never disable it in the common configuration layer, although you can disable it on specific hosts (presentation hosts, for example) by copying the file to a host-specific configuration layer and setting the property to false.

`pollingInterval`

Determines how often the service checks for media content that needs transcoding, in milliseconds. The default value of 30000 means that the service checks for new or modified media content every 30 seconds, and if it finds any, submits it to Elastic Transcoder for transcoding. You might want to shorten this interval, for example:

```
pollingInterval=10000
```

3.2.9 StateChangeNotifier.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/media/services/StateChangeNotifier.properties`**.

You can **optionally** use it to enable notifications of the following events related to media content items:

- Transcoding completed
- Content item published
- Content item unpublished

Notifications are sent to the author and creator of a content item and to the initiators of any of the above events.

> In order for notifications to work, the Content Engine Notes plug-in must be installed.

You only need to set one property in this file to enable notifications:

```
serviceEnabled=true
```

3.2.10 StudioConfig.properties

The full path of this file is **`/etc/escenic/engine/common/com/escenic/webstart/StudioConfig.properties`**. This file is part of a standard Content Engine installation.

Adding Support for New MIME Types

If you need to change Content Studio's MIME type mappings or add support for new MIME types, you can do so by adding a **`property.com.escenic.external-mimetypes`** entry to **`StudioConfig.properties`** (or editing the existing entry if it already exists). You can, for example, add support for MXF format, and associate it with the **`.mxf`** extension with the following setting:

```
property.com.escenic.external-mimetypes={mxf:'application/mxf'}
```

For further information, see MIME Type Mappings.

3.3 Application Configuration Layer Set-up

This set-up task consists of copying example configuration files from the plug-in installation into your publications' application configuration layers and then modifying the copied files to meet your requirements. In detail, you must:

1. Log in to your Content Engine host as the **`escenic`** user.
2. For each publication, create an application configuration layer (if it does not already exists) as follows:

   ```
   $ mkdir -p /etc/escenic/engine/webapp/pub-name/com/escenic
   ```

   where *pub-name* is the name of the publication.
3. Copy all the supplied application configuration layer files as follows:

   ```
   $ cp -r /opt/escenic/engine/plugins/video/misc/siteconfig/webapp/com/escenic/video
    \
   > /etc/escenic/engine/webapp/pub-name/com/escenic
   ```

4. Edit the copied files as described in the following sections.

3.3.1 VideoArticleDecorator.properties

The full path of this file is:

`/etc/escenic/engine/webapp/`*pub-name*`/com/escenic/video/presentation/VideoArticleDecorator.properties`

where *pub-name* is the name of the publication. You only need to set the following property in this file:

`URLGenerator`

How you set this property depends on whether or not you are using CloudFront to distribute your transcoded media content. If you are using CloudFront, then you should set it as follows:

```
URLGenerator=./CloudFrontSignedURLGenerator
```

If you are not using CloudFront then you should set it as follows:

```
URLGenerator=./S3PreSignedURLGenerator
```

Any other properties defined in the file should be left unmodified.

3.3.2 CloudFrontSignedURLGenerator.properties

The full path of this file is:

`/etc/escenic/engine/webapp/`*pub-name*`/com/escenic/video/presentation/CloudFrontSignedURLGenerator.properties`

where *pub-name* is the name of the publication.

You only need to edit this file if you are using CloudFront to distribute your transcoded media content. If you are not using CloudFront, then you should edit **`S3PreSignedURLGenerator.properties`** instead (see section 3.3.3).

`httprotocol`

The protocol you want to be used to access your distributed media files. For example:

```
protocol=http
```

The allowed values are:

- **`http`**
- **`https`**
- **`rtmp`** (for Flash streaming)

`distributionDomain`

The domain name at which the media files are to be published. CloudFront generates a domain name when you create a distribution, consisting of a long id followed by "**`.cloudfront.net`**". You can, however, configure CloudFront to use a domain name of your own, such as:

```
distributionDomain=media.mycompany.com
```

For more information about CloudFront distribution domain names and how to replace them with your own, see Getting Started with CloudFront.

`privateKeyFile`

The path of a file containing a CloudFront private key. For example:

```
privateKeyFile=/etc/escenic/engine/webapp/demo/com/escenic/video/presentation/
cloudfront-pk.der
```

CloudFront generates public/private key pairs that you can use to provide access to restricted content (paid content, for example). The keys are used to create **signed URLs**: automatically generated URLs that allow a customer access to restricted content for a limited period. For more information about this, see Serving Private Content through CloudFront.

> A private key should under no circumstances be shared with any third party, including Escenic and Amazon.

`keyPairId`

The ID of the CloudFront public/private key used to secure your CloudFront content. This is supplied to you along with the private key at the time it is generated.

```
keyPairId=key-pair-id
```

`expiredTime`

The lifetime of the pre-signed URLs generated by CloudFront, specified in minutes. Once this period is over, a pre-signed URL will no longer work and the user will no longer have access to the restricted content. The default value is 30 minutes.

```
expiredTime=30
```

3.3.3 S3PreSignedURLGenerator.properties

The full path of this file is:

`/etc/escenic/engine/webapp/`*pub-name*`/com/escenic/video/presentation/S3PreSignedURLGenerator.properties`

where *pub-name* is the name of the publication.

You only need to edit this file if you are **not** using CloudFront to distribute your transcoded media content. If you **are** using CloudFront, then you should edit `CloudFrontSignedURLGenerator.properties` instead (see section 3.3.2).

`expiredTime`

All content stored in S3 buckets is private. In order to allow users to access content, pre-signed URLs are generated. These URLs have a limited lifetime: once one expires, the content it referenced is no longer accessible. This property specifies the lifetime of the pre-signed URLs, in minutes.

```
expiredTime=10
```

3.4 Transcoder Configuration Files

A transcoder configuration file is an XML file containing details of the Amazon Elastic Transcoder **pipelines** and **presets** available for use with a particular publication or set of publications. A pipeline is a queue for holding transcoding jobs, and a preset is a set of parameters defining how to transcode between two specified formats. Neither pipelines nor presets are **defined** in a transcoder configuration file: they must be defined using Elastic Transcoder, and are then simply referenced from transcoder configuration files. For further information, see the Elastic Transcoder documentation.

You should have already defined the pipelines and presets you are going to need in Elastic Transcoder. You use the transcoder configuration files to specify which pipelines and presets are going to be used by your publications. If all your publications are going to use the same pipelines and presets, then you only need one transcoder configuration file, and you can use the default `/etc/escenic/engine/common/com/escenic/media/aws/transcode-config.xml`:

1. Log in to your Content Engine host as the `escenic` user.
2. Open `/etc/escenic/engine/common/com/escenic/media/aws/transcode-config.xml` for editing.
3. Edit the file to make use of the Elastic Transcoder pipelines and presets you have defined. For further information see section 3.4.1 and section 3.4.2.

If, on the other hand, your publications have different transcoding requirements, you may need to create several different files. Copy the supplied `transcode-config.xml` as many times as required, for example:

1. Log in to your Content Engine host as the `escenic` user.
2. Copy the supplied `transcode-config.xml` as many times as required, for example:

   ```
   $ cd /etc/escenic/engine/common/com/escenic/media/aws/transcode-config.xml
   $ cp transcode-config.xml transcode-config-dailynews.xml
   $ cp transcode-config.xml transcode-config-entertainment.xml
   ```

3. Edit all the files to make use of the Elastic Transcoder pipelines and presets ou have defined. For further information see section 3.4.1 and section 3.4.2.
4. Make sure that the files you create are correctly referenced by the transcoder configurations in your common configuration (see section 3.2.4). Check the `serviceDefinition` property in your `DefaultTranscoderConfig.properties` file and make sure it correctly references your default transcoder configuration file (`transcode-config.xml`). Similarly, check any copies that you have made of `DefaultTranscoderConfig.properties` and make sure

that their **`serviceDefinition`** properties point to the correct transcoder configuration files (**`transcode-config-dailynews.xml`** or **`transcode-config-entertainment.xml`**, for example).

3.4.1 Example Transcoder Configuration File

```
<?xml version="1.0" encoding="UTF-8"?>
<service-definition xmlns="http://xmlns.escenic.com/2012/transcoder-service"
                    xmlns:ui="http://xmlns.escenic.com/2008/interface-hints">
  <pipelines>
    <pipeline default="true" name="low-priority">
      <ui:title>Normal</ui:title>
      <id>pipeline-id</id>
    </pipeline>
    <pipeline name="high-priority">
      <ui:title>High priority</ui:title>
      <id>pipeline-id</id>
    </pipeline>
  </pipelines>
  <presetGroups>
    <presetGroup type="video" name="high-res" default="true">
      <ui:title>High resolution videos</ui:title>
      <presets>
        <preset id="preset-id">
          <thumbnails/>
        </preset>
        <preset id="preset-id"/>
        <preset id="preset-id"/>
      </presets>
    </presetGroup>
    <presetGroup type="video" name="low-res">
      <ui:title>Low resolution videos</ui:title>
      <presets>
        <preset id="preset-id">
          <thumbnails/>
        </preset>
        <preset id="preset-id"/>
        <preset id="preset-id"/>
      </presets>
    </presetGroup>

  </presetGroups>
</service-definition>
```

This example defines a service that provides two pipelines and three preset groups. *pipeline-id* and *preset-id* in a real service definition file would be replaced by Elastic Transcoder IDs - long, unique strings of characters generated when you create a pipeline or preset. Note the following:

- Two pipelines, one of which is for high priority jobs, is a typical set-up. There is no actual difference between the two pipelines. The idea is that most jobs are put in the "normal" pipeline, which may result in long queues during busy periods. Since the "high priority" pipeline is reserved for occasional high priority jobs, long queues should not arise. The prioritization mechanism is therefore dependent on disciplined use.
- A **preset group** is a group of Elastic transcoder presets suitable for a particular purpose: in the example above, **`high-res`** contains the presets needed to convert high-resolution videos to all the

required output formats. What preset groups you create is entirely determined by your particular requirements.

- A preset group must contain either only audio presets or only video presets, and must have a **`type`** attribute indicating what kind of preset group it is.
- One preset group of each type (audio and video) must have the attribute **`default="true"`** set.
- The **`ui:title`** elements are used to specify the names displayed in Content Studio pipeline and preset group selection controls.

3.4.2 transcoder-service

The **`transcoder-service`** schema defines the Escenic Video plug-in transcoder service definition format. You can use it to define the Amazon Elastic transcoder pipelines and presets available for use in a specific publication or group of publications.

Namespace URI

The namespace URI of the **`transcoder-service`** schema is **`http://xmlns.escenic.com/2012/transcoder-service`**.

Root Element

The root of a **`transcoder-service`** file must be a **`service-definition`** element.

3.4.2.1 id

The pipeline ID. This is a number generated by Amazon Elastic Transcoder when a pipeline is created.

Syntax

```
<id>
  text
</id>
```

3.4.2.2 pipeline

Represents an Elastic Transcoder transcoder **pipeline**. A pipeline is a queue for transcoding jobs. For more information see the Elastic Transcoder documentation.

Syntax

```
<pipeline
    default="(true|false)"?
    name="NCName"
  >
  <ui:title>...</ui:title>
    <id>...</id>
</pipeline>
```

Attributes

`default="(true|false)"` (optional)
Whether or not this is the default **`pipeline`**. One **`pipeline`** element should have **`default`** set to **`true`**, and one only.

`name="`*`NCName`*`"`
The name of the `pipeline` element.

3.4.2.3 pipelines

Contains definitions of the pipelines available in this transcoder service.

Syntax

```
<pipelines>
  <pipeline>...</pipeline>+
</pipelines>
```

3.4.2.4 preset

Represents an Elastic Transcoder **preset**. A preset is a template for transcoding media files from one format to another. For more information see the Elastic Transcoder documentation.

Syntax

```
<preset
    id="text"
  >
  <thumbnails/>?
</preset>
```

Attributes

`id="`*`text`*`"`
The preset ID. This is a number generated by Amazon Elastic Transcoder when a preset is created.

3.4.2.5 presetGroup

Defines a preset group. A preset group is a group of **preset**s.

Syntax

```
<presetGroup
    type="(audio|video)"
    name="NCName"
    default="(true|false)"?
  >
  <presets>...</presets>
</presetGroup>
```

Attributes

`type="(audio|video)"`
The media type this `presetGroup` is intended to handle.

Allowed values are:

`audio`
The `presetGroup` is intended to handle audio content.

`video`

The **`presetGroup`** is intended to handle video content.

`name="`*NCName*`"`

The name of the **`presetGroup`** element.

`default="`(true|false)`"` (optional)

Whether or not this is the default **`presetGroup`**. One **`presetGroup`** element should have **`default`** set to **`true`** for each setting of the **`type`** attribute. That is, there should be one default preset group for video, and one default preset group for audio.

3.4.2.6 presetGroups

Contains definitions of the preset groups available in this transcoder service.

Syntax

```
<presetGroups>
  <presetGroup>...</presetGroup>+
</presetGroups>
```

3.4.2.7 presets

A container for **`preset`** elements.

Syntax

```
<presets>
  <preset>...</preset>+
</presets>
```

3.4.2.8 service-definition

The root element of an Escenic Video plug-in transcoder service definition.

Syntax

```
<service-definition>
  <pipelines>...</pipelines>
    <presetGroups>...</presetGroups>
</service-definition>
```

3.4.2.9 thumbnails

Save the **thumbnails** generated by this element's parent preset **`preset`**as keyframes. One **`preset`** element in a video **`presetGroup`** should have a child **`thumbnails`** element. The saved keyframes are added to a relation in video content items, from which the Content Studio user can select an image to use as a poster for the video.

Note that:

- Adding an **`thumbnails`** element to an audio **`preset`** has no meaning.

  Adding an **`thumbnails`** element to a video **`preset`** is only meaningful if the preset has actually been configured to generate thumbnails in Amazon Elastic Transcoder. When you define a preset in Elastic Transcoder, you can specify that it should generate thumbnails at regular intervals (every

60 seconds, for example). You can also specify various characteristics of the generated images (size, format and so on).For details see the Elastic Transcoder documentation

Syntax

```
<thumbnails/>
```

3.4.2.10 ui:title

The display name of a pipeline or preset group. It is used when displaying pipeline/preset group options in Content Studio.

Syntax

```
<ui:title>
  text
</ui:title>
```

3.5 Setting Section Parameters

You can override the default global or publication level **`pipeline`** or **`presetGroup`** settings for part of a publication by setting section parameters in one of the publication's section. The section parameters to use are:

`video.transcoder.config.pipeline`
If specified, this parameter sets the default pipeline to be used for transcoding video content in the current section and all its subsections. The default can be overridden for individual content items if video content types are configured with a pipeline option (see section 3.6.2).

`video.transcoder.config.presetGroup`
If specified, this parameter sets the default preset group to be used for transcoding video content in the current section and all its subsections. The default can be overridden for individual content items if video content types are configured with a preset group option (see section 3.6.2).

3.6 Media Content Type Definition

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 (not currently supported), and both of them can either be **internal** or **external**:

Internal
An internal media 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 a reference to the content. An internal content item is usually created in Content Studio by uploading the video/audio content that is to be published (although it can also be imported in various ways, just like any other content).

External

> External media content types are not supported in the current version of the Video plug-in.

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.

3.6.1 Defining an Internal Video Content Type

An internal video content type must at least have the following:

- A **`media`** child element belonging to the namespace **`http://xmlns.escenic.com/2013/media`**. This element must have a **`type`** attribute set to **`video`**, thereby identifying the content type as a video content type managed by the Video plug-in.
- A parameter called **`com.escenic.article.staging`** set to **`false`**, which 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.
- A **`link field`** for holding a link to a locally stored video file. The **`link`** element must have a child **`relation`** element containing the value **`com.escenic.edit-media`**.
- A **`basic`** field with:
  - **`mime-type`** set to **`application/json`**
  - a **`video`** sub-element with an **`enabled`** 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="internal-video">
  <media xmlns="http://xmlns.escenic.com/2013/media" type="video" enabled="true"
 dropable="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <ui:decorator name="videoArticleDecorator"/>
  <panel name="main">
    <field name="binary" type="link">
      <relation>com.escenic.edit-media</relation>
    </field>
    <field name="video" type="basic" mime-type="application/json">
      <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>
    <mime-type>application/mxf</mime-type>
  </constraints>
</field>
```

Content Studio has a default set of MIME type mappings that cover most commonly used formats. You might, however need to change the default settings or add new MIME types (such as the **`.mxf`** extension used in the example above). For information, on how to do this, see section 3.2.10.

Here is a more realistic version of an internal video content type, this time including a title field, label and so on, but with the important parts highlighted.

```
<content-type name="internal-video">
  <media xmlns="http://xmlns.escenic.com/2013/media" type="video" enabled="true"
 dropable="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <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">
      <constraints>
        <mime-type>video/mpeg</mime-type>
        <mime-type>video/mp4</mime-type>
        <mime-type>application/mxf</mime-type>
      </constraints>
    </field>
    <field name="video" type="basic" mime-type="application/json">
      <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
    </field>
  </panel>
</content-type>
```

3.6.2 Adding Pipeline and Preset Group Options

Adding pipeline and preset group options to media content types enable editorial users to override default pipeline and preset group settings for individual content items.

A pipeline/preset group selection option is a **`field`** element that:

- Has **`mime-type`** set to **`text/plain`**
- Has a child **`pipeline`** or **`preset-group`** element that belongs to the namespace **`http://xmlns.escenic.com/2013/media`** and has the attribute **`enabled ="true"`**

A minimal pipeline option field definition, in other words, would look like this:

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

In practice, however, such a field definition would require Content Studio users to know the names of the available pipelines and enter them correctly. The preferred method is to use a **collection field**. A collection field can be configured to get all currently defined pipeline names from an Atom feed provided by the Video plug-in. The retrieved names are then offered as suggestions when the user starts typing. Here is an example of the recommended way to define a pipeline option field:

```
<field name="pipeline" type="collection" mime-type="text/plain"
       src="escenic/video/config/pipelines" select="content">
```

```
    <ui:label>Pipeline</ui:label>
    <pipeline xmlns="http://xmlns.escenic.com/2013/media" enabled="true"/>
  </field>
```

The **`src`** attribute contains the path of the Video plug-in's pipeline Atom feed and must be set to **`escenic/video/config/pipelines`**. The **`select`** attribute must be set to **`content`**.

A preset group option field can be created in exactly the same way:

```
<field name="preset" type="collection" mime-type="text/plain"
          src="escenic/video/config/presets/video" select="content">
          <ui:label>Preset Group</ui:label>
          <preset-group xmlns="http://xmlns.escenic.com/2013/media" enabled="true"/>
          </field>
```

For video preset groups the **`field`** element's **`src`** attribute must be set to **`escenic/video/config/presets/video`**.

For a detailed description of how collection fields work, see Collection Fields.

3.6.3 Including Key Frames in Video Content Items

Key frames are still images generated from a video clip. They can be hand-selected images of significant points in the video, or they can be automatically generated at fixed intervals. Key frames can be used to represent the video in various contexts - a keyframe image is usually used, for example, to represent the video when it is not being played.

Amazon Elastic Transcoder can be configured to generate key frames (called **`thumbnails`** in the Elastic Transcoder user interface) at fixed intervals. If you want to save the keyframes and associate them with the video to which they belong, then 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">
  <media xmlns="http://xmlns.escenic.com/2013/media" type="video"/>
  <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 store the generated key frames 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="internal-video">
  <media xmlns="http://xmlns.escenic.com/2013/media" type="video" enabled="true"
 dropable="true"/>
  <parameter name="com.escenic.article.staging" value="false"/>
  <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">
      <constraints>
        <mime-type>video/mpeg</mime-type>
        <mime-type>video/mp4</mime-type>
        <mime-type>application/mxf</mime-type>
      </constraints>
    </field>
```

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

For information about configuring Amazon Elastic Transcoder to actually generate the key frames (or thumbnails) you want, see Thumbnail Settings.

4 Using Content Studio

This chapter describes how you can use Content Studio to:

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

4.1 Creating a Video Content Item

How you create a video content item depends upon what kind it is:

Internal

In this case, you upload a video clip from your local machine and it is stored by the Content Engine before being included in your content item.

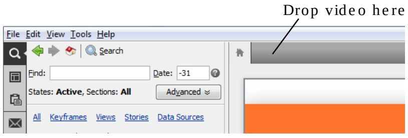
External

External media content types are not supported in the current version of the Video plug-in.

4.1.1 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:



Drop video here

Find:

Date: -31

States: **Active**, Sections: **All**

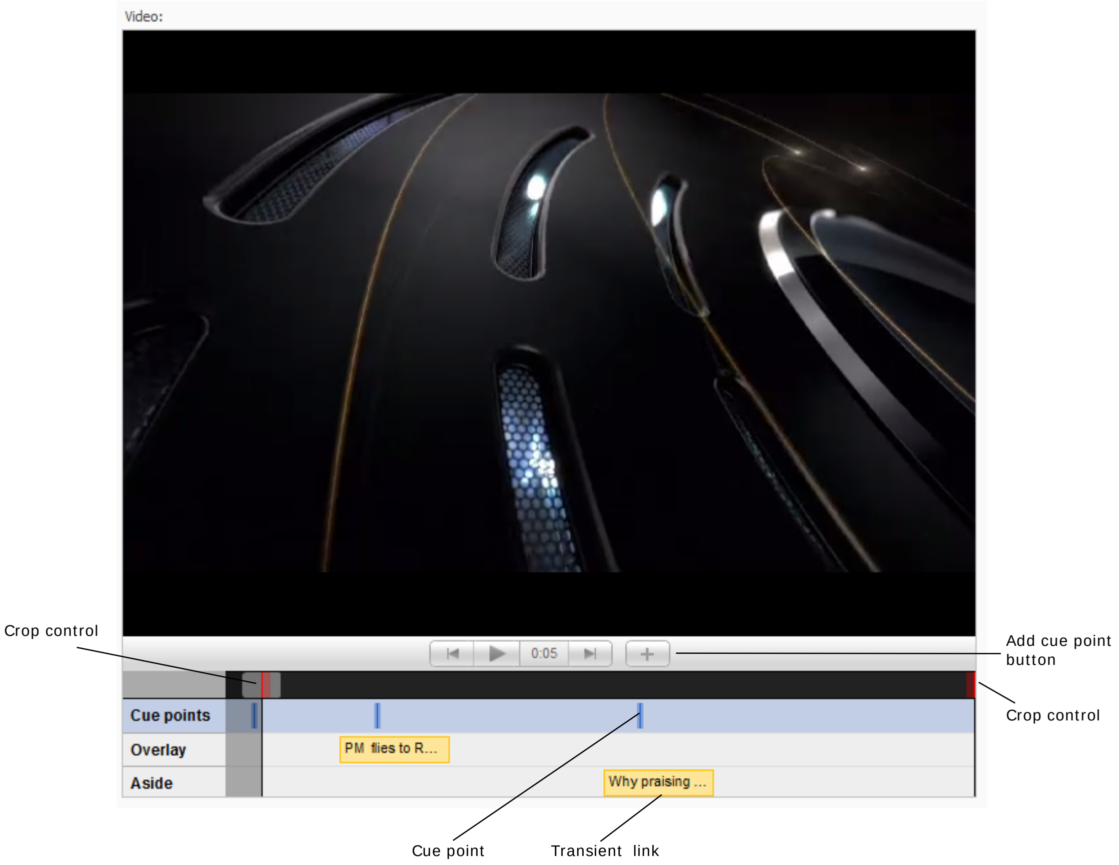
When you drop the video, a new content item of a suitable type is created, and the video is added to it.

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 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.

4.3 Choosing Pipeline and Preset Group

Your media content items **may** contain pipeline and preset group options. The default content types delivered with the Video plug-in contain the following two fields, which you will find on the content items' **Metadata** tabs:

Pipeline

Pipelines are Amazon Elastic Transcoder job queues (see section 1.1 for details). Usually you will have two pipelines to choose from:

Default

Use this queue unless the content item needs to be transcoded and published in a hurry

High Priority

Use this queue only for rush jobs

Preset Group

A preset is an Amazon Elastic Transcoder parameter set defining the output required from a transcoding job (see section 1.1 for details). By choosing a preset group, you determine what formats your video item will be published in. You will usually have a fairly small number of options to choose from, for example:

> These fields may not be available in your media content items (or may be present in some, but not others), and may have different names.

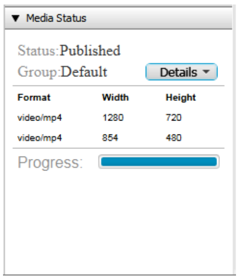
4.4 Publishing Media Content Items

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 content to a variety of formats and encodings for display/playback on different devices and platforms
- Copying the transcoded videos to their published destinations in the network

The different stages in this process are represented by the content item's **media status**, which is displayed in a special **Media Status** section of the Content Studio attributes panel:



Status:Published
Group:Default

Format	Width	Height
video/mp4	1280	720
video/mp4	854	480

Progress:

This section is only present in video content item editors. It shows the media status of the video, the pipeline used to process it 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:

Content Studio state	Media Status
Draft	**Not Transcoded**
Submitted	**Started**

Content Studio state	**Media Status**
	Transcoding **Transcoded**
Approved	**Publishing** **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 clip is processed.

For video 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 transcoding/media publishing actions and media state changes when you move the content item **forward** through the states. No transcoding/media publishing actions are 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 status indicates that transcoding has failed.

Restarted
You may see this status if you change the cropping on an existing video clip that is already published, and publish your changes. It means that the old transcoded clips have been deleted and a new set is being transcoded.

4.5 Notifications

Depending on how your installation is configured, you may receive notifications informing you of various events related to the media publishing process:

- Transcoding completed
- Content item published
- Content item unpublished

Notifications are sent to the author and creator of a content item and to the initiators of any of the above events.

> In order for notifications to work, the Content Engine Notes plug-in must be installed.

5 Automated Media Publishing

This section contains detailed information about how to implement automated media publishing workflows. There are two basic methods you can use to implement such a workflow:

- Web service-based automation, where media content is passed to the Content Engine via its REST web service
- Syndication-based automation, where media content is passed to the Content Engine via its import/export or **syndication** subsystem

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.
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 Amazon Elastic Transcoder for transcoding and key frame generation.

5.1 Web Service-Based Automation

The recommended way to implement an automated media 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. POST the media file you want to publish to the Content Engine web service. The web service has a special fixed URI to which media files and other binary files can be POSTed
2. GET the web service URI for the publication/section to which the content item is to be added.
3. Make an XML document (specifically, an Atom entry) containing all the information required to create the required content item and POST it to the correct URI.

The process of creating a media content item is in fact no different from the process of creating other kinds of binary content items (images, PDFs, office documents and so on). It is described in detail in the following sections of the **Escenic Content Engine Integration Guide**:

- Navigate The Section Hierarchy tells you how to get the upload URL of the section you want to add the content item to.
- Create a Content Item tells you how to create and upload a content item in general.
- Creating Binary Content items provides the additional information you need about uploading binary files to the Content Engine web service.
- Creating Content Items in Different States tells you how to set the state of the content item you create. In order to trigger transcoding and publishing, you need to set the state of the content item to **`published`**.

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 Escenic 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 The Import Service.

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`.

5.3 Working With Video Fields

A video content item must have a **video field** to store various items of video-specific information. The video field usually has the name "video", but if it doesn't you can find out which is the content item's video field by examining the `content-type` resource. The video field is the one that contains a `video` element, like this:

```
<field name="video" type="basic" mime-type="application/json">
  <video xmlns="http://xmlns.escenic.com/2010/video" enabled="true"/>
</field>
```

The video field is used to contain JSON data defining crop points, cue points and transient links, as described in the following sections.

5.3.1 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.

5.3.2 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":cuepoint-values,
    "tracks":track-values
  }
}
```

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 5.3.2.1).

`tracks`
A map containing default transient link definitions for the transient link tracks defined at your installation (see section 5.3.2.2). By default there are two such tracks, called **Overlay** and **Aside**.

5.3.2.1 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.

5.3.2.2 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":cuepoints-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:

`contentID`
An ID for the linked content.

`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.

You can add further tracks in exactly the same way:

```
{
  ...
  "timeline":{
    "cuepoints":tracks-value,
    "tracks":{
      "OverlayDefault":[
        {
          "contentID": "117",
          "startTime": 20.0609756097561,
          "endTime": 23.262195121951216,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        },
       ],
       "AsideDefault":[
        {
          "contentID": "114",
          "startTime": 5.48780487804878,
          "endTime": 8.536585365853657,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        }
       ],
       "ExtraTrack":[
         {
          "contentID": "105",
          "startTime": 3.48780487804878,
          "endTime": 5.536585365853657,
          "title": "DVCPro60_NTSC_5ch_18bit.mxf"
         }
       ]
      }
    }
   }
```

If you want to display a transient link to an external web site rather than to an Escenic content item, you can do so by replacing the `contentID` key/value pair with a `url` key/value pair:

```
"timeline":{
    "cuepoints":tracks-value,
    "tracks":{
      "OverlayDefault":[
        {
          "u
          rl": "http://www.escenic.com/",
          "startTime": 20.0609756097561,
          "endTime": 23.262195121951216,
          "title": "DVCPro50_NTSC_4ch_16bit.mxf"
        },
       ]
      }
    }
   }
```

5.3.3 Complete Video Field Example

Here is a complete JSON example showing the content of a video field:

```
{
  "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": {
      "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"
        }
      ]
    }
  },
  "playback": {
    "in":1.5,
    "out":10.0
  }
}
```

If you download an existing video content item using the web service, or export one, you may see other fields in the data structure that are added by the Content Engine after a content item has been created, but you do not need to supply values for these fields when creating content items.

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:

  ```
  ${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 video's key frames:

  ```
  ${article.relatedElements.KEYFRAMES.items}
  ```

  If a poster frame is selected than the first key frame will be the poster frame.