FS –Media Object

Content

History

Version	Date	Adjustments
A	2006-09-19	First revision. (MMAWI)
B	2008-08-05	Updated with new MediaHandler (merge from FE29 Meteor). (EMATSHA)

1 Introduction

This document specifies the Media Object component.

Basically the Media Object is a container for media related content. The internal format of the media is as it would be stored in a file system.

1.1 Glossary

1.1.1 Media Object

A Media Object (MO) is used to store Media Files (which not imply that they necessarily are stored on a file system, only that the format is the same as a file).

A Media Object may contain information about the properties of the media.

Media Object properties:

- The Content-Type specified as a MIME Media Type as defined in ref. [1].
- File extension, denotes the file format of the Media Object. Examples: wav, txt, mov.
- The size of the media in number of bytes.
- The length of the media. I.e. the number of seconds or pages or both of the media.

The content types supported are: *application*, *image*, *audio*, *text* and *video,* and their relevant subtypes.

Examples on Media Objects:

- A WAV file containing audio in g.711 (PCMU). May have its Content type set to "[*audio/pcmu*]*".*
- A MOV file containing video with amr for the voice track and mpeg for the video track. May have its content type set to [*"audio/amr", "video/mpeg"*].
- A MOV file containing a quicktime video. May have its content type set to ["video/quicktime*"]*
- A text file in Unicode format. May have content type set to [*text/plain;charset=utf-8*]*.*
- A DOC file containing a Word document (may have its content type set to "[*application/msword*]*"*
- A TIF file (may have its Media Properties set to "[*image/tiff*]*")*

1.1.2 MediaFile

MediaFile refers to data formatted as a file in a file system, either as a byte buffer or as a file reference.

2 Function Requirements (Commercial)

Intentionally left blank.

3 Function Specification (Design Related)

3.1 Introduction

3.1.1 Media Object

The Media Object component provides the Media Object, Media Properties, ContentTypeMapper and MediaHandler interfaces. These interfaces are used by its clients to store and retrieve media related contents or to perform basic operations on media object(s).

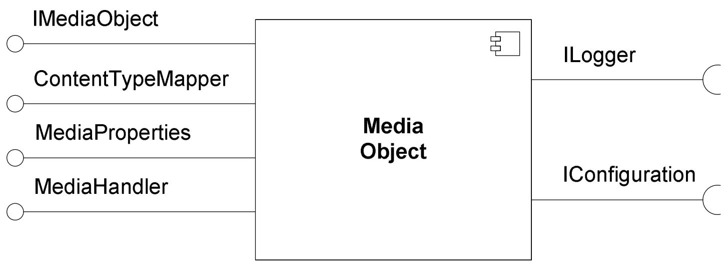

Figure 1 Media object component

The client to Media Object can add a Media File either as a byte buffer or as a file reference. In case a byte buffer is provided, the Media Object internally may store the media either in primary memory or on a regular file system. However the client is unaware of which. If media shall be stored on a file system is depending on the size of the media i.e. at a certain size the media will be stored on a file system. This size is configurable.

3.2 Exported Interfaces

3.2.1 IMediaObject

This interface is used when the client needs to store and retrieve the media. It is possible to store media in a Media Object for example when media need to be recorded or when a Media File from a file system needs to be handled. It is also possible to later on retrieve stored media for example when the media shall be played on an RTP stream or sent as an attachment in an email.
IMediaObject offers the following methods.

3.2.1.1 Methods

3.2.1.1.1 Get Media Properties

MediaProperties getMediaProperties()
GetMediaProperties returns the media properties for the media.

3.2.1.1.2 Set Media Properties

void setMediaProperties(MediaProperties mp)

SetMediaProperties sets the Media Properties of this Media Object.

3.2.1.1.3 Add Media

void addMedia(MediaFile mf)

This method adds a Media File to the Media Object.

3.2.1.1.4 Retrieve Media

MediaFile retrieveMedia()

This method retrieves the Media File from the Media Object.

3.2.1.2 Parameter type description

3.2.1.2.1 MediaProperties

Media Properties contains a list of encodings defined as MIME Media Types for the Media File.

3.2.1.2.2 Media File

MediaFile is either a reference to a media file on a file system or a reference to a buffer in primary memory formatted as it would be such a file.

3.2.2 MediaProperties

This interface is used to handle media properties of a Media File.

3.2.2.1 Methods

3.2.2.1.1 Set ContentType

void setContentType(MimeType contentType)

SetContentType sets the content type provided as a MIME Media Types.

3.2.2.1.2 Get ContentType

MimeType getContentType()

GetContentType returns the MIME Media Type.

3.2.2.1.3 Set FileExtension

void setFileExtension(String fileExtension)

SetFileExtension sets the file extension provided as a String.

3.2.2.1.4 Get FileExtension

String getFileExtension()

GetFileExtension returns the file extension String.

3.2.2.1.5 Set Size

void setSize(long bytes)

SetSize sets the size in number of bytes.

3.2.2.1.6 Get Size

long getSize()

GetSize returns the size in number of bytes.

3.2.3 ContentTypeMapper

3.2.3.1 Methods

3.2.3.1.1 MimeType mapToContentType(MimeType[] codecs)

Maps a set of codecs to a content type. Both the returned content type and the codecs is represented as MIME media types. The codecs is the codecs used in a media. The content type is a general description of the media content.

Example: The codecs "audio/pcmu" and "video/h263" is typically mapped to content type "video/quicktime".

3.2.3.1.2 String mapToFileExtension(MimeType[] codecs)

Maps a set of codecs to a file extension. The codecs is represented as MIME media types. The codecs is the codecs used in a media. The file extension is the extension of the media object if stored on the file system. The purpose of the extension is to represent the file format of the media.

Example: The codecs "audio/pcmu" and "video/h263" is typically mapped to file extension "mov" or "qt".

3.2.3.1.3 MimeType mapToContentType(String fileExtension)

Maps a file extension such as "wav" to a specific content type represented as a MIME media type, such as "audio/wav".

3.2.3.2 Parameter type description

3.2.3.2.1 MimeType

Representation of a MIME Media Type as defined in ref. [1].

3.2.4 MediaHandler

This interface provides methods for manipulating media objects.

3.2.4.1 Methods

3.2.4.1.1 MimeType getMimeType()

Get the MimeType supported by this particular MediaHandler instance. A MediaHandler instance can only handle media objects containing media of a particular MimeType. The MimeType supported is decided upon creation of the MediaHandler instance.

3.2.4.1.2 boolean hasConcatenate()

This method returns true if this particular MediaHandler instance supports the “concatenate()” method. Otherwise false is returned.

3.2.4.1.3 IMediaObject concatenate(IMediaObject mo1, IMediaObject mo2) throws IOException

This method concatenates two different media objects into one newly created media object. The media objects to be concatenated must both contain media of the same MimeType. An IOException is thrown if the concatenation could not be performed

3.3 Imported Interfaces

The Media Object uses the following external interfaces:

- ILogger for logging purposes.
- IConfiguration for configuration

3.4 Functions

3.4.1 Add Video Media Content

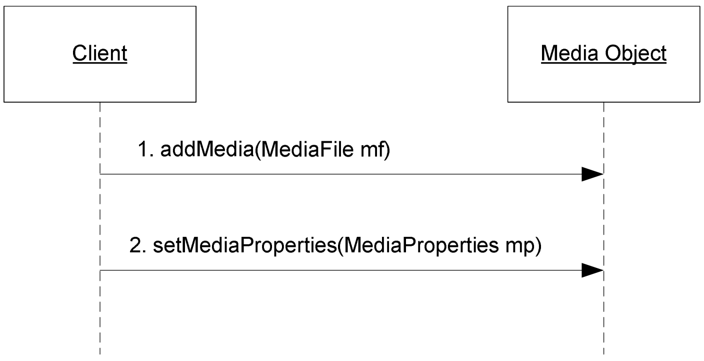
Client
Media Object
1. addMedia(MediaFile mf)
2. setMediaProperties(MediaProperties mp)

Figure 2 Append video media

1. The client adds a Media File to Media Object
2. Optionally the client sets the Media Properties (using its MIME Media Types)

3.4.2 Retrieve Video Media Content

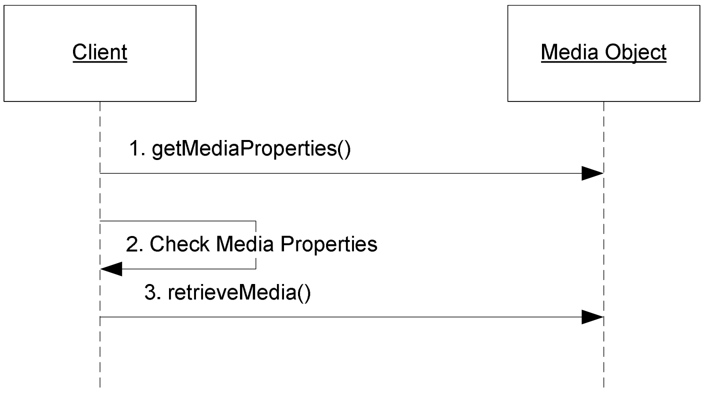

Figure 3 Retrieve video media content

1. The client retrieves the media properties of the media object.
2. The client may check if the content (i.e. its MIME Media Types) is possible to use.
3. The client retrieves the Media File.

3.5 Events

3.5.1 Produced Events

Intentionally left blank

3.5.2 Consumed Events

Intentionally left blank

4 External Operation Conditions

4.1 Configuration

The size of a Media File when to store it to a file system.

5 Capabilities

Intentionally left blank.

6 References

[1] IANA Protocol Number Assignment Services www.iana.org

7 Terminology

Intentionally left blank.